UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Alessandra Bortoletto Garbelotti Hoffmann

Um Modelo para o Controle de Versões de Sistemas para Apoio ao Desenvolvimento Colaborativo através da Web

Dissertação submetida à Universidade Federal de Santa Catarina como parte dos requisitos para a obtenção do grau de Mestre em Ciência da Computação

Orientador : Prof. Vitório Bruno Mazzola, Dr. Inf.

Florianópolis, Novembro-2002

Um Modelo para o Controle de Versões de Sistemas para apoio ao Desenvolvimento Colaborativo através da Web

Alessandra Bortoletto Garbelotti Hoffmann

Esta Dissertação foi julgada adequada para a obtenção do título de Mestre em Ciência da Computação Área de Concentração Sistemas da Computação e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciência da Computação.

_____
Prof. Fernando Álvaro Ostuni Gauthier, Dr.
Coordenador do Curso

Banca Examinadora

_____
Prof. Vitório Bruno Mazzola, Dr.
Presidente da Banca

_____
Prof. Mário Antônio Ribeiro Dantas, Dr.

_____
Prof. Rosesvelter João Coelho da Costa, Dr.

“Há homens que lutam um dia, e são bons.
Há homens que lutam um ano, e são melhores.
Há aqueles que lutam por toda a vida, esses são imprescindíveis.”

(Bertold Brecht)

Oferecimento

Aos meus queridos pais pelo incentivo e carinho sempre recebido. Ao meu esposo pela paciência, amor e bom humor.

Agradecimentos

Ao meu orientador prof. Vitório Bruno Mazzola, que não mediu esforços em "adotar" mais este trabalho.

A minha filha Luiza, que soube esperar o momento adequado para sua vinda e todos aqueles que me auxiliaram em seu cuidado.

Ao colega Everton Coimbra de Araújo pelos incessantes incentivos e ao acadêmico Rodrigo Cesário pelo desenvolvimento dos protótipos das telas modeladas neste trabalho.

SUMÁRIO

Lista de tabelas

Lista de figuras

Resumo

A evolução tecnológica propicia sempre mudanças, e as novas gerações de linguagens para o desenvolvimento propiciam um alto grau de modularização e grande flexibilidade. Concorrentemente as linguagens, surgiram ferramentas para modelagem de dados e de processos, e algumas de auxílio ao controle de versões e desenvolvimento de software em grupo. Com o uso cada vez maior da Internet, o desenvolvimento de projeto de software em grupos (quer seja em correções ou evoluções de sistemas) é cada vez mais comum. A principal proposta do presente trabalho trata do projeto e desenvolvimento de uma ferramenta que engloba o ambiente Worl Wide Web (WWW) da Internet ao gerenciamento das funcionalidades de um desenvolvimento de projetos de software em equipes geograficamente distribuídas. A utilização de uma ferramenta como essa tende a agilizar o processo de construção de software, uma vez que com políticas de manutenção e a recuperação de versões pode haver reuso de código. A facilidade de visualizar os estados e dados sobre as operações realizadas nos arquivos e a comparação entre os mesmos, são outras vantagens para o grupo de desenvolvedores.

Abstract

The technology evolution always provides changes, and the news generations of languages for development of systems propitiated a high modulate degree and great flexibility. Parallelly to the development languages, the tools for modeling of data and processes were appearing, and some of aid to the version control of the software development in group. With the use of Internet, the software project development in groups (whether in updating or systems evolution) is more and more common, there isn't a physical distance anymore. The main proposal of this issue attend of software project development of a tool that integrate of the environment World Wide Web (or WWW) of the Internet, a management of the functionalities of the software project development in people geographicaly distributed. The tendency in a utilisation of the toll is move fast the process of the development, once time that with maintenance politics and the version recover can has source reuse. The facility to visualize the status and data about this operation in files and compararison, are others advantages of the development group.

Capítulo 1

Introdução

O desenvolvimento de um projeto de sistema, desde o seu início, tem normalmente o envolvimento de vários integrantes que formam uma equipe de profissionais de várias áreas, tais como : análise, projeto, desenvolvimento, implementação e treinamento. O que ocorre é que esta "equipe" pode ser a mesma pessoa ou então várias, distribuídas em locais geograficamente distantes, o que hoje é muito comum.

> *"As hardware and software technologies advance, the very nature of the workplace will change".(À medida que as tecnologias de hardware e software avançarem, a própria natureza do espaço de trabalho mudará). (PRESSMANN, 1997).*

Quando se chega a fase de implementação de um projeto e distribuição de versões para avaliação por parte de usuários, algumas empresas deslocam profissionais para o acompanhamento. Nessas situações problemas podem ser detectados e resolvidos, desde que todo o projeto esteja com um profissional, ou então, que o problema seja reportado aos profissionais, com os mesmos conhecimentos, porém situados onde o *código fonte* do projeto em questão esteja ou até mesmo em locais geograficamente distribuído.

A distribuição do código-fonte para um profissional que se desloca para a implementação de um projeto, pode ser viável quando a equipe se resume a um profissional, pois em situação contrária, um mesmo problema poderia ser identificado por mais de um. A manutenção poderia ser feita até um ponto por um profissional e, outro poderia identificar uma outra necessidade e alterá-lo também.

Caso isto ocorra, as seguintes questões surgem: Qual seria a última versão do projeto ? Todos detalhariam as alterações feitas e depois em um único local fariam novamente as atualizações? Após uma atualização,esta seria a válida ? Após isso, uma

nova versão seria enviada para os clientes que já estão com o projeto em produção, porém com versões diferentes ?

A manutenção está também vinculada à modificação de um sistema em conseqüência de alterações no hardware, às modificações para acelerar certos aspectos operacionais do sistema e às modificações em face de alterações dos requisitos do sistema introduzidas pelo usuário.(YOURDON, 1990)

Um sistema de informação deve ser desenvolvido em equipe e a cada componente deve ser atribuída a sua responsabilidade, que a ISO chama de 'Cooperação Mútua'; que também pode ser chamada de Equipe de Desenvolvimento do Projeto de Sistema de Informação. (REZENDE, 1999)

Em todo o processo de evolução do software, é preciso ter controle das versões desenvolvidas. O processo de poder descentralizar esta evolução, fazendo uso da arquitetura e ferramentas utilizadas na Internet, é especialmente importante em sistemas onde os desenvolvedores podem se encontrar em locais diferentes do código fonte ou da documentação do sistema a ser modificado. A demora na manutenção pode ser evitada gerando maior satisfação no ambiente da corporação e redução no custo da manutenção. Este mecanismo de controle do sistema supera as limitações de distância, e não afeta a consistência dos dados.

" Não se pode conservar atualizado um sistema e a documentação a ele associada se essa documentação não estiver correta. Assim, eis o ponto de partida: precisamos garantir que, quando um sistema entrar em operação, todos os documentos a ele estejam completos, consistentes, corretos e atualizados.

Além de nos certificarmos de que os documentos estão corretos, devemos nos assegurar de que existe um mecanismo para executar modificações continuadas nesses documentos. " (YOURDON, 1990)

É especialmente importante ter um processo para controlar as alterações feitas no projeto, sejam elas correções, atualizações ou novas implementações.

Segundo PRESSMANN (1997), os direitos de acesso num ambiente de equipe também podem ser controlados e facilidades de gerenciamento de versões também podem estar disponíveis. Se o conjunto de ferramentas tiver uma capacidade de verificação entre projetos, ele poderá detectar inconsistências entre as contribuições dos vários desenvolvedores.

É difícil, ou até impossível, rastrear a evolução do software através de muitas versões ou lançamentos, se as mudanças não estiverem devidamente documentadas.

Para ROCHA (2001) a diversidade de objetivos exige um processo que garanta qualidade considerando os diferentes aspectos do projeto. Sendo que os projetos com boa documentação podem ser avaliados quanto a sua qualidade com maior precisão.

1.1 Objetivos

Desenvolver um modelo de uma ferramenta de apoio ao desenvolvimento de projeto de sistemas em um ambiente colaborativo distribuído, utilizando como ambiente a Internet.

1.1.1 Objetivos Específicos

Para que o objetivo principal seja alcançado, deverão ser atingidos os seguintes objetivos específicos:

?? Revisão dos princípios e conceitos sobre Software Configuration Management (SCM) , o qual dá apoio ao gerenciamento e desenvolvimento de software.

- ?? Pesquisa dos conceitos sobre sistemas colaborativos seus requisitos para ambiente de desenvolvimento colaborativo e suas arquiteturas, procurando estabelecer a que melhor ofereça segurança e desempenho.
- ?? Realização de uma revisão bibliográfica sobre a tecnologia WWW(World Web Wide) visando verificar as possibilidades do uso desta última ser utilizada em aplicações de controle de versões de sistema, sendo este um ambiente colaborativo distribuído.
- ?? Análise de ferramentas disponíveis relacionadas ao controle de versões do desenvolvimento de projetos, buscando verificar a sua aplicabilidade;
- ?? Modelar um sistema de controle de versões utilizando o ambiente colaborativo distribuído(Internet);

1.2 Limitações do estudo

Devido ao rápido crescimento e uso de novas tecnologias, agregado à pesquisas efetuadas em todo o mundo à cerca do proposto, a busca de informação será efetuada, em sua maior parte, através do próprio objeto de estudo, a Internet.

1.3 Estrutura Trabalho

O desenvolvimento deste trabalho estará apoiado em uma seqüência de capítulos, necessários ao entendimento do proposto. Este primeiro capítulo é composto pela motivação do estudo, os objetivos e organização deste trabalho.

O segundo capítulo, demonstra a importância de um controle de versões para a Engenharia de Software, tratando assim Gestão de Configuração de Software.

O terceiro e quarto capítulos apresentam um estudo bibliográfico, sobre conceitos, artigos e opiniões sobre desenvolvimento colaborativo e a plataforma World Wide Web (WWW) respectivamente.

Na seqüência, no quinto capítulo é realizada uma análise entre algumas ferramentas disponíveis que realizam o controle de versões em um desenvolvimento de projeto.

Após os estudos realizados, o sexto capítulo descreve um modelo de um sistema de controle de versões que utiliza a Internet como interface, e procura contemplar todos os aspectos positivos estudados no capítulo anterior.

O sétimo capítulo apresenta as conclusões e perspectivas de trabalhos futuros.

Capítulo 2

GESTÃO DE CONFIGURAÇÕES DE SOFTWARE

É muito comum, no decorrer do desenvolvimento de um software, efetuarem-se mudanças para melhoria do software ou para correção de seus componentes e relacionamentos entre eles. Ainda, é evidente que um sistema não acaba quando o executável é gerado. Modificações podem ser necessárias, resultado de falhas, acréscimos de informações ou atualizações em razão de mudança na política do mercado.

Portanto, diante dessas mudanças, a necessidade de manter uma cópia de tudo que é feito e modificado torna-se importante, pois as alterações descontroladas podem gerar erros e perdas de informações. Para isso, essas cópias devem ser gerenciadas e controladas de forma que seja possível a recuperação das mesmas quando necessário. O SCM, ou Software Configuration Management, é uma atividade abrangente aplicada em todo o processo de Engenharia de Software, e que é responsável por gerenciar a evolução de sistemas de software grandes e complexos (PRESSMANN, 1995). O SCM identifica, controla, faz a auditoria e relata as modificações que inevitavelmente ocorrem quando o software está sendo desenvolvido e mesmo depois que ele é distribuído aos clientes.

Somente se houver um processo organizado que controle as modificações, as inconsistências, ou perda de informações, serão evitadas, garantindo qualidade na vida do projeto.

Na Engenharia de Software, a documentação, o modelo e o próprio código (fonte e executáveis) devem ser controlados visando a integridade dos produtos de software através do controle do processo de desenvolvimento. Assim, não importa a complexidade do projeto, o controle de versões é importante também para que não haja desperdício de trabalho. Se houver esse controle, versões antigas podem ser

aproveitadas em outro projeto, numa consulta, ou simplesmente para evitar que venham ser utilizadas de forma incorreta.

Acredita-se que a qualidade de Software está diretamente ligada à qualidade em que o Software é desenvolvido. É impossível atestar que determinada aplicação é adequada, sem que se conheça e controle suas etapas de desenvolvimento. Tal é a sua importância, que as principais iniciativas internacionais de avaliação e de certificação de qualidade, entre elas ISO 9000[1] e CMM[2], estabelecem a implantação de SCM como um dos requisitos básicos para a obtenção de certificados de qualidade de processos de software.

LOZANO (2002) em seu artigo diz que : Infelizmente, a maioria das empresas, especialmente aquelas que têm no desenvolvimento de software a sua atividade-fim, não utiliza a maioria dessas ferramentas. Segundo a Carnegie Mellon University, que criou o Capability Maturity Model para o software (www.sei.cmu.edu/cmm/cmm.html), mais de 80% das empresas de software (ou departamentos de TI de outros tipos de empresas) ainda estão no nível 1, que o CMM define como "caótico".

"Mesmo não conhecendo os procedimentos de gerência de configuração de software, uma empresa pode adotar práticas que de alguma maneira pode auxiliar na gerência e na qualidade dos produtos desenvolvidos. Por outro lado, uma empresa que não conhece os procedimentos de gerência de configuração de software e tampouco adota certas práticas no desenvolvimento e manutenção de seus produtos, certamente mais cedo ou mais tarde defrontará com os problemas relacionados com a qualidade de seus produtos desenvolvidos e de seus serviços prestados em relação à manutenção." (GCS, 1999).

---

[1] ISO 9000 – série de certificação- conjunto de normas que tratam de sistemas da qualidade do processo de software. Demonstra que o sistema de Qualidade da Organização é efetivo (ROCHA, 1998. p.14)

[2] CMM- *Capability Maturity Model* – definido no Software Engineering Institute (SEI) – Carnegie Mellon University . define o nível de maturidade de uma empresa quanto ao desenvolvimento dos processos (ROCHA 1998, p.15)

2.1 Conceito

O SCM – Software Configuration Management, pode ser visto como uma disciplina de apoio ao gerenciamento e desenvolvimento de software. No caso de apoio ao gerenciamento, o SCM gerencia o controle de alterações dos produtos de software fazendo a identificação dos componentes do produto e suas versões, o controle de alterações (pelo estabelecimento de procedimentos a serem seguidos quando uma alteração é realizada), a contabilidade de status (registrando os status dos componentes e pedidos de alteração), a análise e a revisão (garantia de qualidade das funções dos componentes para preservar a consistência do produto). Já no caso de apoio ao desenvolvimento, o SCM fornece funções que auxiliam os programadores na realização coordenada de alterações nos produtos de software. Além disso, o SCM auxilia os desenvolvedores com a composição dos produtos de software a serem controlados, registrando assim, suas revisões (versões subseqüentes, seriais) e variantes (versões paralelas, independentes), mantém a consistência entre componentes dependentes, reconstrói configurações de software previamente gravadas, constrói objetos derivados (código compilado e executável) de suas fontes (programa texto) e edifica novas configurações baseadas nas descrições de suas propriedades.

A Configuração de sistema é a documentação das características do hardware e software que compõem um sistema, de modo a controlar sistematicamente suas mudanças e manter sua integridade durante o ciclo de vida do sistema.

Gerenciamento de configuração (SCM) envolve uma coleção e manutenção de dados importantes de hardware e software de sistemas que estão sendo utilizados (ALLAN, 1997).

Na Figura 2.1. é representado um modelo espacial de uma configuração , onde pode ser visualizado o hardware e algumas pessoas em típico ambiente comercial, onde o software está incluído. O que não é mostrado é a documentação e o trabalho que constitui toda a configuração.

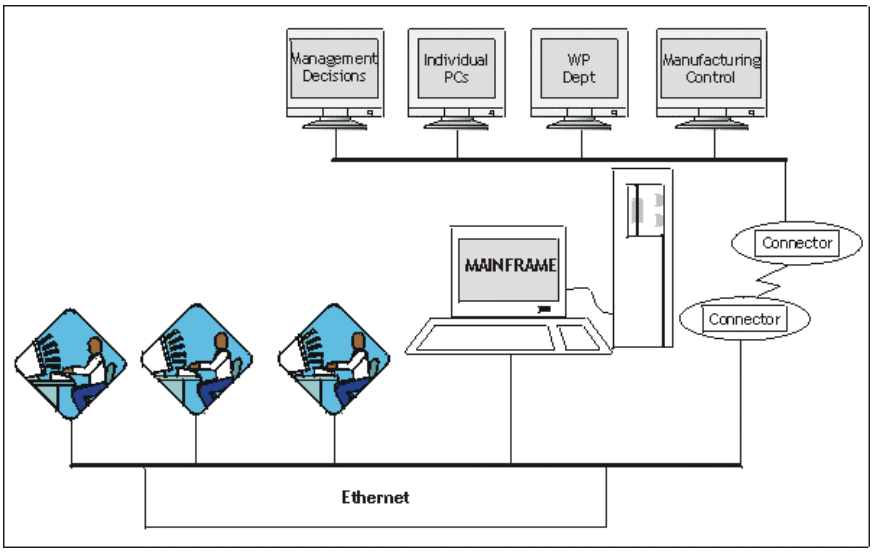


Figura 2.1 – Modelo Espacial adaptado de ALLAN (1997)

O controle de versão é realizado com diferentes propósitos. Uma versão destinada a substituir seu predecessor é chamada revisão (versão serial). Além disso, uma revisão pode ser definida como sendo a forma mais simples de criar novas versões a partir de uma modificação na versão anterior. Dessa forma, as versões formam uma lista encadeada que pode ser referida como uma cadeia de revisões (MARK, 2000).

Geralmente, uma nova revisão é gerada quando se torna necessário fazer alguma melhoria na versão anterior, prover algum aumento de funcionalidade, corrigir problemas, adaptar a versão anterior a algum outro ambiente, etc. Por outro lado, segundo SOARES (2001), versões que não substituem seu predecessor são chamadas variantes (versões paralelas).

Finalmente, as versões podem também ser mantidas para apoiar a cooperação. Nesse caso, múltiplos desenvolvedores trabalham em paralelo em diferentes versões e cada desenvolvedor opera em um workspace que contém as versões criadas e usadas. Para isso, algumas políticas de cooperação são necessárias para controlar quando as versões devem ser exportadas de um workspace ou importadas para dentro de um workspace (CONRADI, 1998).

Para RADDING (1999), uma ferramenta de gerenciamento de configuração de software pode auxiliar as empresas a dominar ou controlar os problemas dispersos nas equipes de desenvolvimento.

2.2 REPRESENTAÇÃO

Com base nos conceitos sobre modelos de versão de software descritos até aqui, serão ilustradas algumas formas de representação de organização de versões. Existem várias formas de representar o espaço da versão, mas a maioria dos sistemas SCM faz essa representação através de grafos.

Conforme descrito em SOARES (2001), um grafo de versão consiste de nós e arestas que correspondem às versões e seus relacionamentos, respectivamente. No caso mais simples (organização unidimensional), um grafo de versão consiste de um conjunto de versões conectadas por relacionamentos de um único tipo chamado sucessores. Um grafo de versão deste tipo representa a evolução histórica de um item, onde "v2 é sucessor de v1" significa que "v2 derivou de v1" através de alguma alteração aplicada em v1. Os grafos de versão podem ser representados de diferentes formas, como mostra a Figura 2.2. Na maioria dos casos, as versões podem estar organizadas em uma seqüência de revisões como está ilustrado na Figura 2.2 (A). Na representação através de árvores, Figura 2.2 (B), os sucessores de versões que não são folhas podem ser criados, por exemplo, de forma a manter as versões anteriores já distribuídas. A Figura 2.2 (C) representa um grafo acíclico, onde uma versão pode ter múltiplos predecessores (isso ocorre quando é realizada uma operação merging).

Um grafo de versão em organização bidimensional é composto por branches (ramificações laterais), sendo que cada branch consiste de uma seqüência de revisões. Nesse caso, no mínimo dois relacionamentos são necessários: sucessores (dentro de uma branch) e descendentes (entre as branches), como ilustra a Figura 2.3.

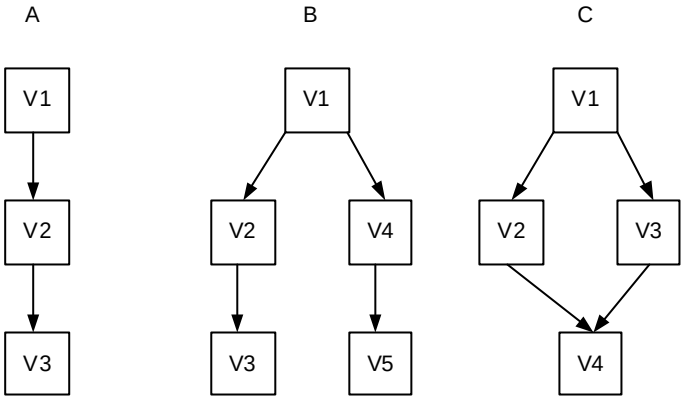


Figura 2.2 - organização unidimensional

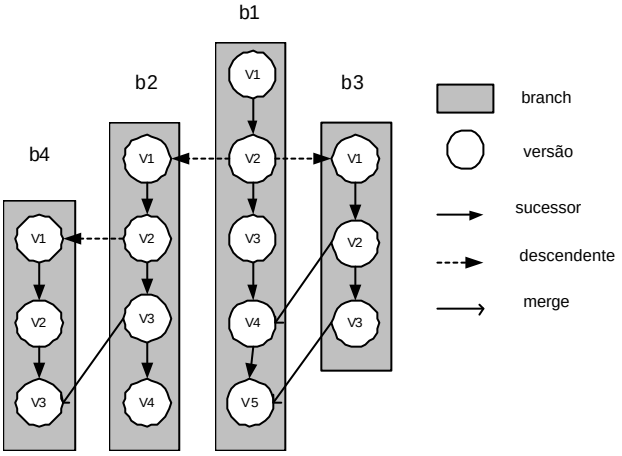


Figura 2.3 - organização bidimensional

As versões de um componente de software são muitas vezes organizadas em uma árvore histórica (ancestral), a qual tem uma versão principal (root) que representa a primeira versão de um componente de software. A árvore histórica inicial é simples, consiste somente de uma branch, chamada tronco, mas com o prosseguimento do desenvolvimento, branches laterais podem surgir. As branches podem surgir, na maioria das vezes, nas seguintes situações segundo SOARES (2001):

?? no desenvolvimento simultâneo entre múltiplos usuários;

?? no desenvolvimento distribuído ;

?? quando versões anteriores ainda precisam ser melhoradas (alteradas);

Ainda SOARES (2001) relata em seus estudos que a tarefa essencial do SCM é armazenar a árvore histórica de versões eficientemente e, para isso, várias técnicas têm sido propostas (BIELIKOVÁ, 1999[3]).

Para economizar espaço nos dispositivos de armazenamento secundários (como disco, por exemplo) a maioria das ferramentas de controle de versão armazena versões na forma de deltas. Delta é o conjunto de diferenças entre duas versões subseqüentes de um mesmo arquivo (CONRADI, 1998). Em outras palavras, delta pode ser definido como uma seqüência de comandos de edição que transformam uma string em outra (TICHY, 1985).

A idéia principal para economizar espaço de armazenamento é a seguinte: se uma versão for derivada de uma outra versão (ou seja, um sucessor na árvore histórica) então as duas versões provavelmente têm uma grande parte em comum e um pequeno número de diferenças. Portanto, para economizar espaço, uma versão é armazenada completa e a outra em forma de delta, ou seja, armazena-se somente as diferenças de conteúdo. Durante a análise de armazenamento dos deltas, dois aspectos são importantes:

- como gerar um delta entre dois arquivos;
- como aplicar o delta na árvore histórica de versão, ou seja, qual versão armazenar de forma completa e em qual versão aplicar o delta.

Atualmente, as técnicas de armazenamento de deltas mais conhecidas são o delta reverse e o delta forward. Para gerar um delta entre dois arquivos, um algoritmo para isolação da seqüência comum entre os dois é usado (TICHY, 1985).

---

3 Bieliková,M.,"Space-efficient version storage", site disponível em: wysiwyg://63/http://www.dcs.elf.stuba.sk/~bielik/scm/delta.htm. Visitado em agosto de1999 por SOARES (2001)

2.3 Objetivos

Segundo PAULA (2001) os objetivos do SCM são os seguintes :

- ?? Identificar resultados dos projetos que serão submetidos ao SCM;
- ?? Controlar alterações e versões destes itens;
- ?? Manter a integridade durante todo o projeto.

Ainda visa garantir que :

- ?? Todos os resultados intermediários e finais, associados a marcos importantes de todos os projetos sejam colocados e controlados em bases de dados de Gestão de Configurações;
- ?? Organização dos itens em linhas de base;
- ?? Todas as alterações a itens das linhas de base sejam controladas e checadas;
- ?? A recuperação do histórico dos Itens de Configuração de Software;
- ?? A recuperação de versões por membro da equipe ou interessados de todos os itens de Configuração de Software.

2.4- Linha de Base (Baseline)

Uma linha de base ou baseline consiste em um ou mais Itens de Configuração de Software que permite controlar as mudanças que ocorrem nos resultados de um projeto, durante a execução deste. Esses Itens de Configuração de Software são previstos em padrões e no Plano de Qualidade do Software de um projeto. A linha de base deve ser um conjunto consistente, ou seja, as especificações, desenhos, descrições de teste, código e documentação de usuário devem estar todos revisados, estando no mesmo nível.

"Uma baseline é um documento ou produto que tenha sido formalmente revisado e combinado podendo servir como base para favorecer o desenvolvimento. " (IEEE 610-1990; IEEE 828-1990)

Cada linha de base corrente, que é a última linha de base criada (no projeto), corresponde a um projeto, e, para que seja alterada é preciso que algumas etapas de formalização (análise, aprovação e registro) sejam preenchidas.

2.5 Planejamento

BOUNDS (2001) , define 10 elementos "chaves" para implantar o Gerenciamento de Configuração (SCM) em uma organização. São o planejamento, processo, cultura, pessoas, produto, automação, gerência , o projeto, o sistema e a aprovação. Os primeiros sete elementos relatam o problema e a solução, os outros três são os resultados dos sete anteriores. O projeto, sistema e a aprovação são pontos principais da solução SCM. Todos os dez elementos são mostrados na Figura 2.4 .

- Planejamento (Planning): definir todos os procedimentos e políticas que devem ser adotadas para o projeto de CM ;
- Processo (Process): descrição atual do processo de CM e o nível de controle desejado quando o processo de CM for implantado;
- Cultura (Culture): entender os vários tipos de culturas que existem dentro de uma organização a fim de encontrar a solução de CM;
- Pessoas (People): definir todas as responsabilidades e tarefas das pessoas envolvidas durante a implementação do processo de CM;
- Produto (Product): determinar que produto e quais as partes deste que serão controladas pelo CM tais como código fonte, componentes, documentos e ou ferramentas;
- Automação (Automation): definir os requisitos para a funcionalidade do sistema de CM tais como gerenciamento de objetos e modelo do sistema;

?? Gerenciamento (Management): decisões gerenciais associadas com a solução de CM tais como comprar ou desenvolver a ferramenta. Quando e como começar a automatizar o sistema.

?? Projeto (CM plan) : este é o projeto que será implementado e que o CM precisa. Todos os procedimentos, políticas, cronogramas, responsabilidades, entre outros devem estar definidos nesse elemento.

?? Sistema CM (CM system): é a escolha da ferramenta para automatizar o processo de CM.

?? Aprovação (CM adoption strategy): aprovação das estratégias utilizadas pela organização.

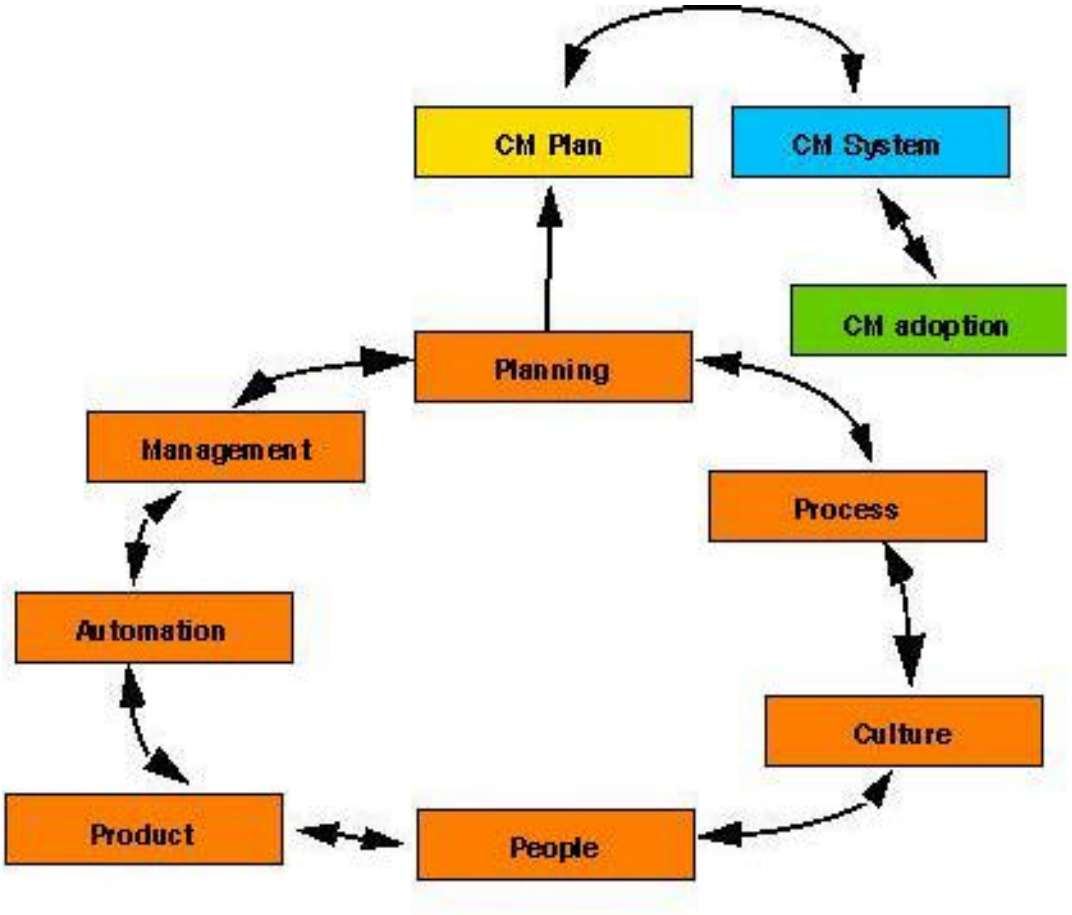


Figura 2.4 . Elementos do Gerenciamento de Configurações[4]

[4] Figura retirada de BOUNDS,Nadine M. e Susan Dart. **CM Plans: The Beginning to your CM Solution**. The Software Engineering Institute (SEI), 2001.

2.6 – Estrutura de Identificação em Projeto de Software

A divisão é realizada através de pastas e subpastas. Na pasta consta o projeto e nas subpastas o diretório de cada desenvolvedor. No caso de componentes de código, deve estar consistente como Modelo de Desenho do Software.

Um mesmo item pode pertencer a mais de uma linha de base.Todo item de Gestão de Configurações, é identificado pelos seguintes campos:

- ?? Número de revisão (interno – não da versão final do produto);
- ?? Nome do item (nome do arquivo);
- ?? Zero ou mais nomes das subpastas a que pertence;
- ?? Nome do projeto a que pertence;
- ?? Nome da linha de base a que pertence.

2.7 – Estrutura de Identificação dos Produtos de Software

PAULA (2001), recomenda que cada projeto tenha o mesmo nome do respectivo produto.

É classificado segundo o esquema de identificação das versões:

- ?? Projeto de desenvolvimento novo
- ?? Projeto de evolução - acréscimo de novas funções sendo estas uma quantidade significativa
- ?? Projeto de melhoria - acréscimo de pequenas melhorias
- ?? Projeto de transporte - implantação em novas plataformas de operação (SO)

2.8 Bibliotecas de Configuração

"A Gestão de Configurações envolve uma grande quantidade de processamento de informações, não importando o tamanho do projeto. Dessa forma, a gestão de uma Biblioteca de Configurações de Software é viável apenas quando realizada por uma ferramenta automatizada." (PAULA, 2001)

É necessário existir duas Bibliotecas de Gestão de Configuração de Software, são elas:

- Biblioteca de Desenvolvimento de Software
- Biblioteca de Manutenção de Software

Na Biblioteca de Desenvolvimento de Software, a última linha de base corresponde a primeira linha de base na Biblioteca de Manutenção de Software, servindo de fonte dos itens que serão reutilizados no desenvolvimento de uma nova versão. Sempre que uma nova versão esteja sendo elaborada, as alterações devem ser repassadas ao gerente do projeto.

2.9 CONCLUSÃO

Com a globalização do mercado, a exigência de qualidade dos produtos de software em conformidade com normas e padrões de avaliação (ISO 9000 e CMM entre outras) é cada vez maior. Assim, conhecer a Gestão de Configuração (SCM) e, adotar ferramentas para gerenciar o controle do desenvolvimento de sistemas é necessário para se obter melhores resultados de qualidade no processo de software.

Mas, para a adoção de ferramentas existe um planejamento que deve ser observado para que a qualidade seja alcançada. As estruturas de identificação do projeto de software são necessárias para organização. Enquanto a identificação do produto de software procura estabelecer normas para o entendimento da numeração inserida quando realizada sua nova versão, a identificação do Projeto de Software realiza a divisão do projeto.

Capítulo 3

Sistemas Colaborativos e o desenvolvimento de software

O cenário de desenvolvimento na década de 60 e 70 era em ambientes computacionais dominados pelos mainframes, que concentravam as tarefas da corporação. O acesso as informações eram feitas através de terminais ligados ao servidor central.

Hoje o cenário é outro, e com a evolução da tecnologia tais como o PC, transmissão de dados e redes de computadores, o desenvolvimento de sistemas distribuídos fez surgir o modelo cliente/servidor.

Os sistemas computacionais colaborativos auxiliam realizações conjuntas através de redes de computadores. Nesses sistemas, vários usuários compartilham informações e dados de uma ou mais aplicações, com o propósito de efetuar um objetivo comum. Esses usuários podem estar localizados todos em um mesmo local, bem como distribuídos remotamente.

Nesses tipos de sistemas há mecanismos que permitem coordenar atividades de muitos usuários além de oferecerem outros que definem propriedade a determinados dados, determinam as limitações e atribuições correspondentes e suas permissões, e ,estabelecem relacionamentos entre os mesmos.

Dessa forma, é possível afirmar que os sistemas colaborativos definem a base para muitas aplicações em redes, como teleconferências, jogos, chats de conversação, sistemas de suporte ao ensino a distância e o desenvolvimento de aplicações em equipe.

Segundo HILLS (1997), “a colaboração gera um produto que é maior que a soma de suas partes. Algumas descobertas em ciência, medicina e outros campos resultaram de esforços de dois ou mais colaboradores”.

Portanto percebe-se que as perspectivas de aplicação de sistemas colaborativos são bastante amplas e tendem a se difundir ainda mais com a evolução tecnológica, no sentido de ampliar a performance computacional e a popularização das redes de auto desempenho.

No desenvolvimento de um software, normalmente estão envolvidos vários especialistas, sendo que o trabalho em equipe é fundamental. Nesse sentido um sistema que tenha um controle sobre as alterações que ocorrerão no desenvolvimento deste é de fundamental importância pois tarefas já realizadas poderão ser reutilizadas reduzindo tempo, bem como reduzir despesas com deslocamento de membros da equipe já que o propósito é o controle de versões de sistema com ambiente WWW, podendo ser utilizado em lugares remotos.

Este capítulo aborda os conceitos sobre trabalho colaborativo e sua contribuição no desenvolvimento específico de sistemas.

3.1 Visão Conceitual de Trabalho Colaborativo

Existem vários conceitos para Sistemas Colaborativos e que são aceitos como referência de um único objeto de estudo tais como: trabalho cooperativo, Groupware, Trabalho Cooperativo Suportado por Computador – CSCW, computação colaborativa (collaborative computing) , Sistema de Suporte a Grupo - GSS (group support system) que estão descritos conforme abordagem de cada autor. Sendo assim, são essenciais para o entendimento que se pretende neste trabalho.

Para RIZZO (2002), colaboração é a integração de diferentes Tecnologias em uma simples aplicação ou ambiente para compartilhar e gerenciar informações.

> *"...Collaboration—at least in part—is the integration of many different technologies into a single application or environment to facilitate information sharing and information managemen..."*

"Um sistema colaborativo é uma ferramenta que auxilia as pessoas a trabalharem em conjunto com mais facilidade, pois a comunicação, coordenação e colaboração são essenciais. Podem ainda ser denominados como collaborative computing (computação colaborativa), group support system (GSS - Sistema de Suporte a Grupos) ou ainda Groupware". (INFORMATIVO, 2000)

Para HILLS (1997), Groupware é uma ferramenta que ajuda as pessoas a trabalhar juntas com mais facilidade ou eficiência, permitindo que se comuniquem, coordenem e elaborem.

CSCW -Trabalho Cooperativo Suportado por Computador - é definido como a disciplina de pesquisa para o estudo das técnicas e metodologias de trabalho em grupo e das formas como a tecnologia pode auxiliar este trabalho. Portanto, enquanto CSCW é a pesquisa na área do trabalho em grupo e como os computadores podem apoiá-lo, Groupware é empregado para designar a tecnologia (hardware e software) gerada pela pesquisa em CSCW. Desta forma, correio eletrônico, teleconferência, suporte à decisão e editores de texto colaborativos são exemplos de Groupware.

Ainda, observou-se que os termos cooperação e colaboração não devem ser usados como sinônimos . A cooperação seria realizada pela divisão de trabalho entre os participantes, como uma atividade em que cada pessoa é responsável por uma porção da resolução do problema, sendo a atividade distribuída hierarquicamente. A colaboração se caracterizaria pelo engajamento mútuo dos participantes em um esforço coordenado para juntos resolverem o problema.

3.2 Funções do trabalho colaborativo

Ainda para HILLS (1997) um groupware consiste em hardware e software em uma rede, com as seguintes finalidades :

- ?? ajudar duas ou mais pessoas a trabalharem juntas;
- ?? permitir o compartilhamento de experiências e conhecimento;
- ?? automatizar suas atividades;
- ?? ajudar a criar uma memória organizacional;
- ?? possibilitar superar incompatibilidades entre geografia e tempo.

Além de minimizar custos referentes a viagens e tornar a comunicação mais rápida, mais clara e mais persuasiva, um groupware ajuda as pessoas a : compartilhar informações (comunicação), coordenarem suas atuações individuais e a colaboração em equipe.

Em RADDING (1999), Charlotte Payne declara que dar mobilidade aos projetos, faz com que não seja preciso deslocar desenvolvedores, onde os custos com mudança de ambiente - considerando todo o transtorno que poderia causar em toda uma família – podem ser reduzidos, além dos problemas familiares com o funcionário.

3.3 Aplicações do Trabalho Colaborativo

A noção de interdependência nas tarefas é um ponto importante e serve para diferenciar colaboração de outros tipos de tarefas em grupo, como a interação, por exemplo.

> *Considere a diferença entre dirigir em um comboio e dirigir no trânsito de uma cidade. No primeiro caso, os motoristas têm algum sistema de comunicação pré-estabelecido e um objetivo comum que depende do sucesso dos outros motoristas nas suas tarefas (todos os*

> *carros do comboio devem chegar ao destino; se algum precisar de ajuda, os outros certamente o socorrerão). Isso caracteriza uma colaboração. Por outro lado, ao dirigir no trânsito de uma cidade não há colaboração, há apenas interação entre os motoristas, pois não há nenhum planejamento prévio das tarefas e o sucesso de cada motorista em atingir seu objetivo não depende do sucesso dos demais motoristas.(GROSZ ,1996)*

Isto pode ser considerado uma dependência positiva, como característica do trabalho colaborativo, pois cada participante precisa que o trabalho do outro seja bem sucedido. O caso oposto é quando várias pessoas simplesmente compartilham um recurso. Nesta situação, é preciso haver uma certa coordenação entre as tarefas, para que o trabalho de um apenas não atrapalhe o trabalho dos outros. (SCHIMIDT ,1992).

HILLS (1997) classifica as aplicações de acordo com a maneira como as pessoas utilizam cada ferramenta:

Trabalho em conjunto : Neste caso, as pessoas podem estar juntas no mesmo lugar ou em lugares diferentes, mas devem estar trabalhando ao mesmo tempo. O propósito destas ferramentas é tornar as reuniões mais eficientes e aumentar a colaboração.

São exemplos Conferência de voz, Vídeo Conferência, Sistemas eletrônicos de reunião, Whiteboarding[5] e ferramentas de bate-papo.

Trabalho Individual : Não importa quando ou onde a pessoa esteja trabalhando. O propósito destas ferramentas é reduzir e substituir as reuniões (com horários pré-determinados), propiciando ainda assim formas de colaboração, pois permitem aos colaboradores trabalharem de onde quer que estejam e no horário que melhor lhes convier.

---

[5] Nome atualmente utilizado na Internet para ferramentas mais conhecidas como conferência de dados. (HILLS, 1997)

Alguns exemplos: Correio eletrônico, conferência e discussões, banco de informações, escrita em grupo ou ferramenta de edição de documentos compartilhados e ferramentas de Workflow (fluxo de trabalho).

Os dados da Tabela 3.1 indicam que 34,3% das empresas brasileiras adotam uma prática onde cada integrante da equipe trabalha em cópia própria do arquivo, independente dos demais membros. Isto indica pelo menos uma preocupação de que o desenvolvimento em equipe ou as atualizações dos arquivos seja feito em arquivos separados. Por outro lado, 65,7% das empresas adotam a prática de se trabalhar numa única cópia do arquivo, compartilhada por todos os elementos da equipe. (GCS ,1999)

Tabela 3.1 . Distribuição das Empresas Quanto à Abordagem Adotada no Desenvolvimento de Software em Equipe.

Abordagem	*Nº de empresas*	*%*
Cada integrante da equipe trabalha,independente dos demais, em cópia própria do arquivo	**121**	**34,3**
Uma única cópia do arquivo é compartilhada por todos os elementos da equipe	**232**	**65,7**
Total	***353***	***100***

Fonte : ***Pesquisa GCS -Gerência de Configuração de Software. Fundação Centro Tecnológico para Informática - CTI / IC . Programa de Qualidade e Produtividade em Software – PQPS. Área de Tecnologia para Produção de Software – ATPS. ? ?Fundação CTI 1999***

3.4 Formas de Colaboração

RAPOSO (1999), classifica os Sistemas Colaborativos levando em consideração dois fatores: o modo de interação e a distribuição geográfica dos usuários. (ELLIS, 1991; FLUCKIGER, 1995).

Com relação ao modo de interação, sistemas colaborativos podem ser :

1. ***Assíncrona* :** Você não precisa estar presente para participar, ou seja em diferentes momentos.
   Ex : E-mail, a Internet e Intranet são formas de comunicação assíncrona.

2. ***Síncrona*** : Quando o trabalho é realizado ao mesmo tempo (tempo real).
   Ex: Chats, Vídeo Conferência

Quanto à localização geográfica dos usuários, um sistema pode ser:

a) Presencial : usuários presentes fisicamente no mesmo local;
b) ou Não-presencial : usuários em locais diferentes.

Para WILSON (1994), além da classificação espaço-tempo definida na Figura 3.1, os sistemas colaborativos também podem ser divididos em quatro categorias de acordo com o tipo de aplicação, mas não são exclusivas, ou seja, um sistema pode possuir característica de uma ou mais categoria. As categorias são descritas abaixo.

Comunicação : ferramentas de e-mail, chats e vídeo-conferência.

Espaço de trabalho compartilhado: provêem uma área de trabalho comum, onde dois ou mais participantes podem trabalhar conjuntamente. Exemplos de ferramentas nesta categoria são os whiteboards compartilhados e os editores multiusuários (nos quais vários usuários têm acesso a um documento sendo editado, e podem estar simultaneamente alterando partes diferentes dele).

Informações compartilhadas: permitem que duas ou mais pessoas armazenem, acessem e manipulem informação compartilhada. Bancos de dados compartilhados são exemplos deste tipo de ferramenta. Sob esta óptica, a Web pode ser vista como ferramenta colaborativa, pois permite que as pessoas compartilhem informação.

Suporte à atividade em grupo: provêem facilidades para atividades específicas do trabalho em grupo, tais como agendamento de reuniões, brainstorming, auxílio na

tomada de decisões, argumentação, votação, etc. Este tipo de ferramenta, também conhecido como GSS (Group Support System), tem como objetivo aumentar a produtividade de reuniões, seja acelerando o processo de tomar decisões ou melhorando a qualidade das decisões resultantes .(ELLIS ,1991).

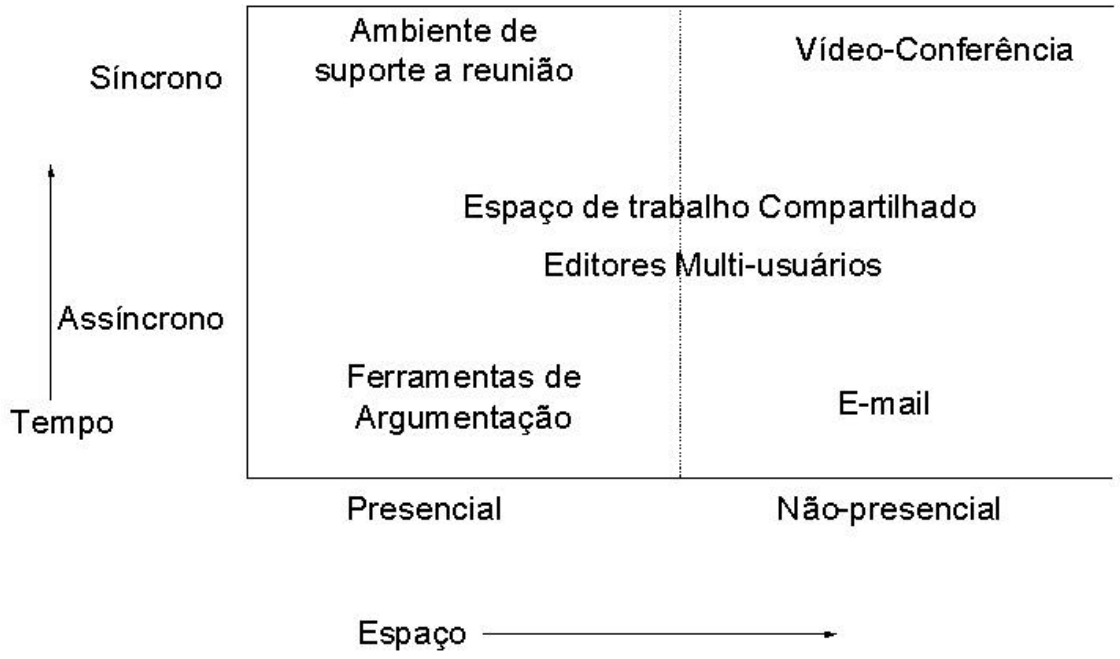


Figura 3.1 Classificação de sistemas colaborativos - adaptado de FLUCKGER (1995)

3.5 Arquitetura em Ambientes Colaborativos

Encontramos conforme RAPOSO (1999) descreve, as seguintes arquiteturas para sistemas colaborativos :

?? ***Ambiente com dados compartilhados centralizados***: (Figura 3.2) há uma base de dados localizada em servidor central, compartilhada por todos os usuários. Todas as alterações de estado realizadas pelos usuários devem ser comunicadas apenas ao servidor, que redistribui a mensagem a todos os participantes. Como a base de dados é definida como um único recurso, apenas um usuário pode alterá-la de cada vez. A vantagem neste caso é a inconsistência devido à base de dados ser centralizada. No

entanto, o servidor é muito sobrecarregado e tende a se tornar o fator limitante do sistema. Além disso, o tráfego de mensagens é muito grande (as mensagens são enviadas primeiro para o servidor e então distribuídas para os participantes).

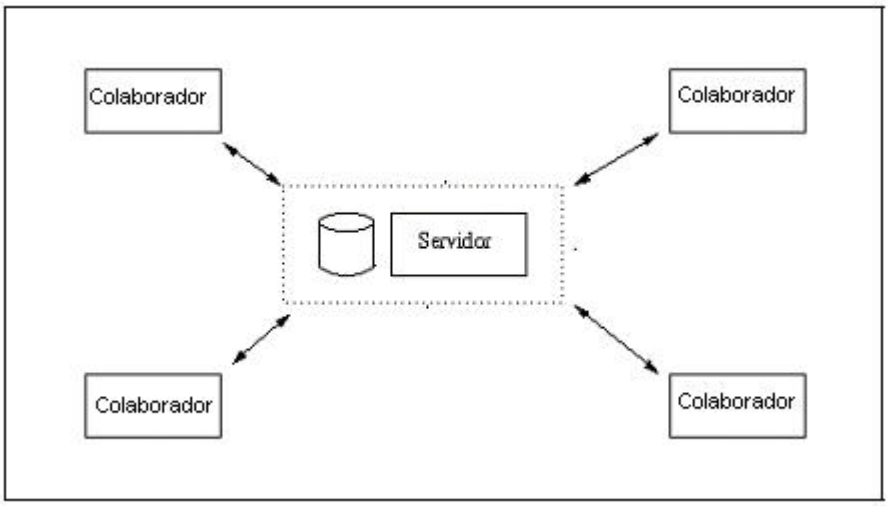


Figura 3.2: Ambiente com dados compartilhados centralizados.

?? ***Ambiente com dados compartilhados distribuídos, atualização ponto a ponto***: (Figura 3.3) os dados são replicados parcial ou totalmente em cada participante nesse tipo de ambiente. Não existe um servidor central, como nos casos anteriores e as alterações são comunicadas aos participantes via *multicast*[6]. A grande vantagem desse tipo de ambiente é não ter o servidor limitando o desempenho do sistema. No entanto, existem problemas de escalabilidade devido aos custos para a manutenção da confiabilidade e consistência de dados.

ORAM (2001), reafirma que em um sistema com arquitetura totalmente distribuída, não só cada host[7] é um participante igual, como não há hosts com recursos especiais ou papéis administrativos.

Além do mais, a comunicação ponto a ponto é mais frágil no aspecto da segurança (todos os participantes têm os mesmos privilégios) e tende a sobrecarregar os participantes com mensagens.

---

[6] transmissão para um número de receptores ou nós, com um endereço em cada mensagem para indicar o nó desejado. (Dicionário Eletrônico Michaelis, 2001)

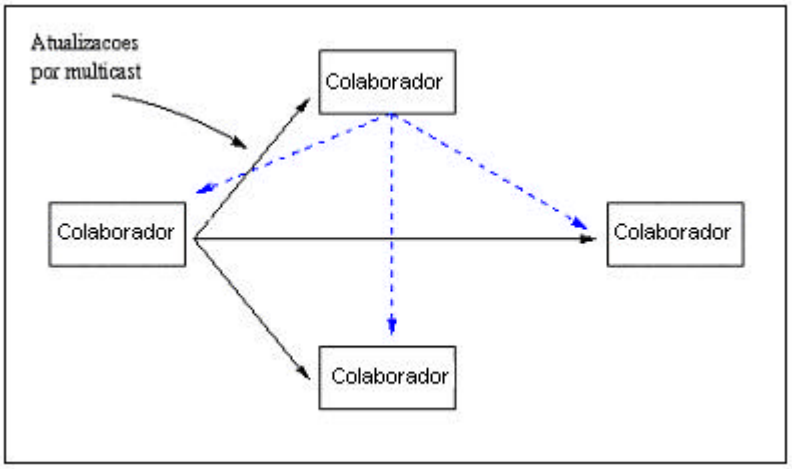


Figura 3.3: Ambiente com dados compartilhados distribuídos, atualização ponto a ponto.

DELGADO (1998) observa que uma série de aspectos caracteriza sistemas distribuídos, e quando a computação colaborativa se junta a estes sistemas, outros fatores surgem alterando de forma significativa os aspectos já existentes.

Ambientes colaborativos distribuídos permitem que um grupo de usuários ou aplicações, dispersos geograficamente, possa usar computadores e diversas mídias (texto, audio, vídeo e imagens) para resolver problemas conjuntamente. (YAVATKAR ,1994)

"A descentralização gera uma área totalmente nova de falhas relacionadas à rede: irresponsabilidade, sincronização incorreta dos dados, etc. Os projetistas de sistemas precisam pesar o poder dos modelos peer-to-peer em relação às complicações e limitações de sistemas descentralizados". (ORAM, 2001)

?? **Ambiente com dados compartilhados distribuídos, cliente-servidor**: variante do modelo cliente-servidor, onde os dados estão divididos entre os participantes e a comunicação é mediada por servidor central. Esse tipo de ambiente representado na

---

[7] cada computador é visto como principal

Figura 3.4, supera os problemas característicos da comunicação ponto a ponto (dificuldade de consistência, por exemplo). No entanto, a presença de um servidor pode servir como fator limitante do sistema, especialmente em ambientes com número elevado de usuários (os sistemas existentes usam técnicas para reduzir a comunicação entre os participantes e conseqüentemente o uso deste servidor).

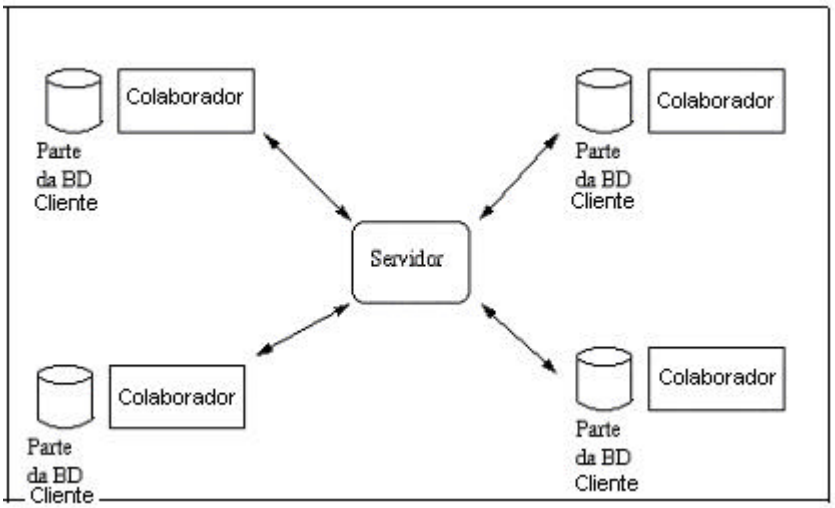


Figura 3.4 Ambiente com dados compartilhados distribuídos, cliente-servidor

?? **Ambiente homogêneo replicado**: o estado de cada colaborador é iniciado a partir de uma base de dados homogênea contendo as informações do projeto. Cada colaborador recebe cópia do projeto inicial e só precisa ser comunicado das alterações efetuadas pelos outros membros. Nesse tipo de ambiente é grande a quantidade de pequenas mensagens transmitidas. Ele também é relativamente pouco flexível e, com o tempo, tem tendência ao aparecimento de inconsistências entre os colaboradores devido à perda de mensagens, situação demonstrada na Figura 3.5.

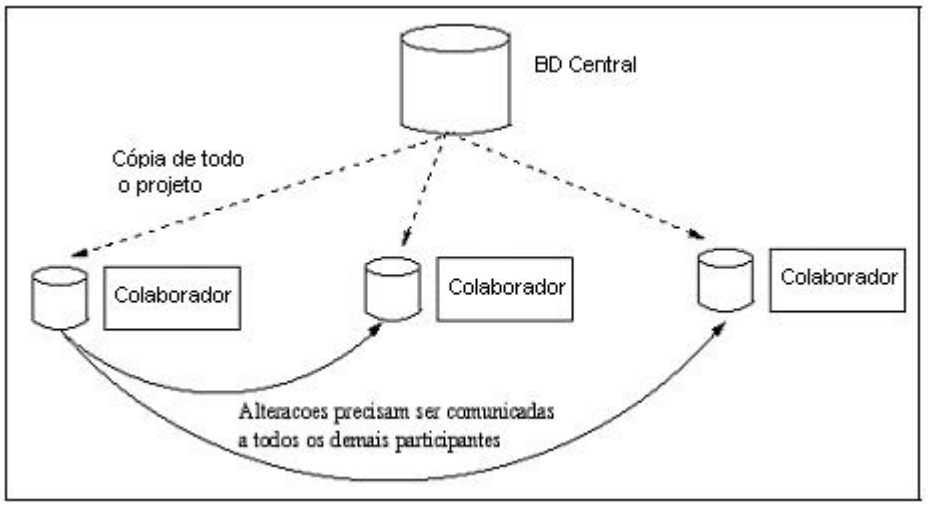


Figura 3.5 Ambiente homogêneo replicado

3.6 Requisitos para Ambientes Colaborativos

Um dos principais aspectos que caracteriza um ambiente colaborativo é o fato de suportarem redes de computadores. A Internet e a Intranet[8] oferecem um ambiente distribuído perfeito para servir de suporte ao desenvolvimento de sistemas colaborativos. Um importante ganho que pode ser obtido com a Internet está no fato dela ser inerentemente assíncrona, exigindo assim um controle de versões quando abordado o desenvolvimento de sistemas em um ambiente colaborativo distribuído.

A comunicação é parte essencial da colaboração. Sua importância pode ser verificada nas classificações apresentadas na seção 3.4, pois comunicação estabelece a dimensão temporal dos sistemas (comunicação síncrona ou assíncrona) e também define uma classe de aplicações quanto a localização geográfica de um colaborador (presencial ou não-presencial). A comunicação também é o núcleo do trabalho de articulação das atividades colaborativas.

O desafio que CSCW apresenta às tecnologias de comunicação é como tornar a comunicação entre usuários remotos tão eficientes quanto à interação face a face

[8] termo de uso geral que significa Internet interna, utilizada por algumas organizações e empresas. Os dados e comunicação nesta abordagem são protegidos do mundo exterior. (HILLS,1997)

(ELLIS, 1991). Na verdade, sabe-se que a tecnologia não substitui a interação face a face, mas apresenta meio alternativo de comunicação, que pode ser mais apropriado em certas situações. O desafio é aplicar as combinações tecnológicas mais adequadas para cada tipo de interação .

Para KLING (1991), um tipo de comunicação durante muito tempo negligenciado nas pesquisas de groupware foi a comunicação informal. Ao contrário da comunicação informal, a comunicação formal segue um modelo de conversação pré-estabelecido, que pode ser representado por alguma ferramenta formal (redes de Petri, entre outras). No entanto, é sabido que nem os padrões de comunicação e nem a estrutura dos grupos são estáveis no decorrer de um trabalho colaborativo, o que torna estes modelos de conversação bastante limitados e de pouco "realismo social". Por esta razão, tem sido cada vez mais comum o desenvolvimento de sistemas oferecendo oportunidades para a realização de encontros informais à distância, tanto para a comunicação entre indivíduos quanto para a comunicação entre grupos.

Além da diferenciação entre comunicação formal e informal, também é possível classificá-la em comunicação direta e indireta. A comunicação direta ou explícita, se dá através da transmissão de texto, gestos, vídeo e/ou áudio entre os participantes (é o que se chama normalmente de comunicação). A comunicação indireta ou implícita, se dá através do objeto de trabalho (por exemplo, um texto ou imagem compartilhados). (REINHARD ,1994)

Assim, há necessidade de flexibilidade para o modelo de comunicação utilizado no trabalho colaborativo. Generalizando esta idéia, a flexibilidade de uma aplicação colaborativa pode ser analisada do ponto de vista técnico, funcional, do campo de aplicação e também do ponto de vista social.

Flexibilidade técnica está relacionada à portabilidade e capacidade de adaptação do sistema nos diversos ambientes computacionais. Uma aplicação é considerada flexível tecnicamente se permitir que o trabalho colaborativo seja realizado em diferentes plataformas de hardware, sistemas operacionais, interfaces gráficas, com

vários formatos de áudio e vídeo e diferentes dispositivos de entrada e saída. (REINHARD, 1994)

DELGADO (1998), ressalta que um ambiente colaborativo via de regra utiliza diversos programas de diferentes fabricantes e procedências de forma combinada para atingir algum objetivo ou executar alguma tarefa. Isto exige que estes programas possam trocar informações e se comunicar entre si de forma uniforme. Para isto devem adotar padrões de arquitetura de informação e distribuição que torne a sua utilização independente de sua procedência e independentes da plataforma em que serão utilizados. Isto caracteriza o que se denomina interoperabilidade e é geralmente atingida pela adoção de protocolos abertos .

Flexibilidade funcional está relacionada à classificação espaço-tempo. Um sistema que se enquadre nas várias categorias e permita o "chaveamento" entre os vários modos de interação e localização geográfica dos usuários é mais flexível no aspecto funcional.

Flexibilidade do ponto de vista do campo de aplicação está relacionada a um sistema que suporte mais tipos de aplicações (de comunicação, de espaço de trabalho compartilhado, etc) é mais flexível sob este aspecto.

A flexibilidade do ponto de vista social é bem mais complexa, tendo sido tema de vários trabalhos que tentam alertar os projetistas de sistemas colaborativos sobre as características dinâmicas e pouco previsíveis do trabalho em grupo. (KLING, 1991; SCHMIDT, 1991).

Desta forma para serem flexíveis, as aplicações colaborativas não devem impor padrões de trabalho ou comunicação pré-estabelecidos. Na verdade, elas devem prover facilidades que permitam aos usuários interpretar e explorar estes modelos formais, mas deverá sempre caber ao usuário a decisão de usá-los, modificá-los ou rejeitá-los. (SCHMIDT, 1991).

Com relação à arquitetura utilizada, PENG (1993) define que os sistemas podem ser centralizados, replicados ou híbridos. A configuração centralizada segue o modelo cliente-servidor, onde todos os participantes só podem se comunicar com o servidor central, que realiza o processamento necessário e retransmite as mensagens. Dentre as vantagens da configuração centralizada estão sua relativa simplicidade de implementação, facilidade para garantir consistência no estado da aplicação (existe apenas uma cópia executando no servidor), além de permitir clientes mais simples e evitar os complexos algoritmos distribuídos.(SHIRMOHARMMADI, 1998)

A configuração replicada ou distribuída executa uma cópia da aplicação em cada máquina. As vantagens desta configuração incluem a geração de menos tráfego na rede, já que as mensagens não precisam passar pelo servidor, e uma maior escalabilidade do sistema (o servidor se sobrecarregaria acima de um determinado número de clientes). Tentando unir as vantagens destas duas últimas configurações, surgiu a configuração híbrida. Nesta configuração, cada cliente executa uma cópia (completa ou parcial) da aplicação e há também a presença de um servidor central, responsável por tarefas como a garantia de consistência entre os clientes, sincronização e tratamento de colisão de eventos (por exemplo, dois usuários querendo editar a mesma parte de um documento).

3.7 Vantagens e Desvantagens do Trabalho Colaborativo

Quando uma empresa assume o trabalho colaborativo como técnica para melhorar a produtividade de uma tarefa, o que primeiro vem à mente é a redução de gastos. Mas como toda técnica, existem vantagens e desvantagens, pois o trabalho colaborativo envolve comunicação entre pessoas e uma efetiva organização da equipe.

3.7.1 Vantagens

CANIZARES (2001), define como vantagens principais da colaboração :

- Implementação mais eficiente de projetos;
- Desenvolvimento técnico-profissional dos envolvidos;

- ?? Comunicação favorecida;
- ?? Eliminação de duplicidade de esforços;
- ?? Avaliação mais acurada nas necessidades;
- ?? Aumento na disponibilidade de recursos;

Com os membros de uma equipe colaborando e tendo como principal objetivo o desenvolvimento pleno de um projeto, o crescimento individual é favorecido, já que idéias e soluções são compartilhadas e reutilizadas. Desta forma reduz-se o tempo quando algum problema semelhante é detectado pois sua solução pode já ter sido resolvida.

Observando o lado do cliente, este também será beneficiado, já que o esforço da equipe é principalmente com vista na sua satisfação.

Ainda, é possível citar :

- ?? Redução de gastos com deslocamento de membros da equipe;
- ?? Reutilização de soluções e código;

3.7.2 Desvantagens

Considerando para o escopo deste trabalho - sistema de controle de versões - as desvantagens apontadas por CANIZARES (2001):

- ?? Queda brutal no nível de cooperação quando ocorrem crises no grupo ou programa;
- ?? Segurança das informações - Um mesmo trabalho não deve estar sendo realizado por mais de um colaborador ao mesmo tempo;

Se um colaborador não confia em seus parceiros, não estará aberto e receptivo a novas idéias nem haverá motivação para ele compartilhar recursos e problemas com o grupo.

"Durante crises que podem ocorrer internamente nas organizações envolvidas, ou dentro da própria parceria, a cooperação entre os membros pode decrescer sensivelmente, ameaçando a continuidade do programa em comum. As organizações participantes particulares,comunitárias e governamentais – todas elas às vezes se defrontam com mudanças em sua estrutura, cortes no orçamento, alterações administrativas e outros imprevistos que podem afetar seu compromisso de cooperação ." (CANIZARES, 2001)

3.8 Conclusão

Com o advento das redes de computadores, principalmente a *Internet*, a comunicação passou a ser bidirecional, ou seja, a colaboração entre usuários. Tem-se observado, porém, que para que um desenvolvedor se sinta mais engajado no projeto é necessário que ele possa interagir, procurando realizar os trabalhos, trocar idéias e ajudar os outros colegas da equipe. Assim está se buscando em *groupware*, ferramentas e metodologias que possam ser apropriadas para o desenvolvimento em equipe. Um ambiente de desenvolvimento colaborativo pode, porém, diminuir o tempo de resolução de um problema. Isso porque as idéias já encontradas poderão ser reutilizadas.

Para HILLS (1997), As ferramentas de groupware permitirão que equipes interfuncionais compartilhem conhecimento e idéias para serem mais inovadoras.

A equipe geograficamente dispersa pode se beneficiar com o contato e o trabalho de forma mais eficiente e produtiva, onde as decisões podem ser tomadas com muito mais rapidez.

Capítulo 4

Ambiente World Wide Web em sistemas colaborativos

Apesar da Internet ter nascido no início dos anos 70 e que os principais protocolos responsáveis por seu funcionamento ainda são os mesmos daquela década, a popularização da Internet ocorreu praticamente no início dos anos 90, com o aparecimento da tecnologia e das aplicações WWW- World Wide Web (MAZZOLA,2001).

As grandes vantagens em se desenvolver aplicações disponibilizadas via WWW estão associadas à fácil acessibilidade: as aplicações ficam disponíveis para uma ampla gama de usuários da Web e elas podem ser acessadas de praticamente qualquer lugar (de casa, do trabalho, ou mesmo em trânsito, através da computação móvel), possibilitando a comunicação à longas distâncias. A comunicação realizada por outros tipos de mídias tais como o telefone, permite a comunicação "um para um".O rádio, a televisão permite "um para muitos", mas a Internet permite que a comunicação seja "muitos para muitos" , que agrega as duas anteriores, caracterizando-a com um uso mais amplo.

Somada a estas vantagens, ainda existe a portabilidade[9] das aplicações Web, onde a disponibilidade de diversas informações e serviços é cada vez mais comum.

Este capítulo tem como objetivo verificar a Internet, através de seus recursos, como a ferramenta possível para a implantação do modelo proposto por este trabalho, pois gradativamente ela está se tornando um meio usual de troca de informações de forma rápida, rompendo barreiras geográficas de espaço e tempo, e permitindo o compartilhamento de informações, cooperação e comunicação em tempo real.

---

[9] independência de plataforma

4.1 A Internet

A Internet (InterNetwork System – *Sistema de Interconexão de Rede de* Comunicação), considerada a rede das redes de comunicação, pode permitir a comunicação e compartilhamento de recursos e dados com pessoas em seu bairro ou ao redor do mundo.

"O modelo de rede dos aplicativos de usuário – não apenas o uso de banda larga, mas também seus métodos de endereçamento e comunicação com outras máquinas – mudou significativamente com o surgimento da Internet comercial e o advento de milhões de usuários domésticos na década de 90". (ORAM, 2001)

O crescimento, desde o seu surgimento na metade da década de 60 até hoje , foi muito rápido . O TCP/IP (**Transmission Control Protocol over Internet Protocol**) é o conjunto de protocolo que é sua base de sustentação.(MAZZOLA ,2001)

4.1.1 O Surgimento da World Wide Web

A World Wide Web surgiu praticamente como resultado da iniciativa de um pesquisador inglês Tim Berners-Lee, do Centro Europeu de Pesquisa em Energia Nuclear (CERN, Suíça), que desenvolveu, em 1989, um trabalho de especificação de protocolos e programas que foram os embriões da Web.

A partir daí, as aplicações da Internet que eram praticamente baseadas em comandos de linha e alguns poucos clientes escritos utilizando interfaces gráficas ainda rudimentares tornaram-se, a maior parte delas, baseadas no uso de navegadores (browsers), capazes de exibir, não apenas informação em formato de texto, mas informação em formatos cada vez mais diversificados, como gráficos, vídeo, som e voz, dentre outros.

Com o surgimento de linguagens de programação orientadas à Internet, a interatividade das páginas Web com os usuários tornou-se um outro ponto de fundamental importância, no uso da Web como tecnologia para a implementação de sistemas de informações e outras aplicações. (MAZZOLA ,2001)

4.2 Tecnologia e Arquitetura utilizadas na World Wide Web – WWW

A popularização da Internet como meio de comunicação de massa tornou-se possível apenas com o surgimento do ambiente WWW- World Wide Web, uma vez que, até então, o uso das ferramentas Internet com base em interfaces de comandos de linha não era favorável a usuários que não tivessem habilidades avançadas no uso de um computador.

O passo inicial para o surgimento da então chamada "Internet Colorida" veio com o trabalho realizado por Tim Berners-Lee, que especificou um ambiente para localização de relatórios de pesquisa e demais documentos utilizando técnicas de hipertexto e hipermídia, sendo que o segundo termo foi introduzido também por Ted Nelson para definir um documento mais abrangente do que um documento de hipertexto, abrindo então a possibilidade de introdução de outros formatos de informação, como aqueles já mencionados na seção anterior.

A Figura 4.1 apresenta o esquema original apresentado por Tim Berners-Lee, que utilizava o conceito de um servidor específico para o atendimento às solicitações de acesso aos documentos. Os principais elementos de tal ambiente foram, então:

?? máquinas clientes, caracterizadas por computadores dos usuários em busca de documentos; estes computadores poderiam ser de plataformas diversas (UNIX, DOS, Windows, Macintosh, etc.);

?? servidores hipermídia, caracterizados por máquinas de grande porte, contendo grandes capacidades de armazenamento e processamento, para atender de forma eficiente às solicitações das máquinas clientes; estes servidores poderiam manter referências a documentos residentes em outros servidores, para permitir uma melhor distribuição da informação e uma maior facilidade na localização dos documentos;

?? "browsers", que seriam programas de aplicação específicos, com capacidade de comunicação com os servidores e capazes de enviar solicitações para busca de documentos.

Em seu projeto original, Berners-Lee já apresentava a preocupação de lidar não apenas com documentos que fossem gerados especificamente para o novo sistema, mas também com documentos e informações que já estivessem disponíveis nos diversos sistemas de informação pelo mundo.

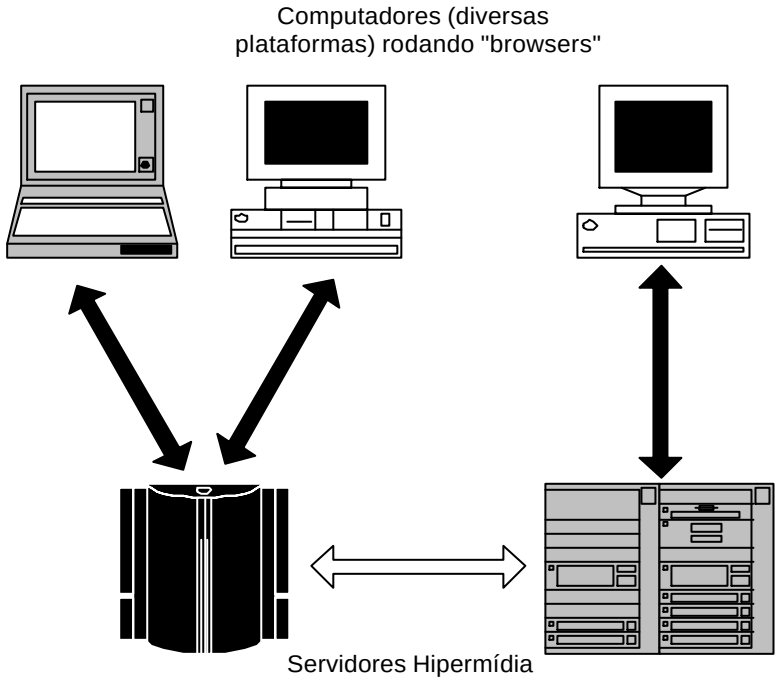


Figura 4.1 – Esquema de funcionamento do sistema de acesso via hipermídia.

Algumas bases de dados referenciadas por Berners-Lee foram grupos de discussão (newsgroups), manuais UNIX, Centro de Documentação do CERN, catálogos telefônicos, bases de dados em geral.

O termo World Wide Web surgiu pouco tempo após a publicação dos trabalhos de Tim Berners-Lee, onde o último W, de Web, fazia referência a uma "teia" de servidores e clientes que poderiam comunicar-se e tornar acessíveis as informações armazenadas nos diversos servidores em todo o mundo. Por isso o nome "teia de alcance mundial".

4.2.1 Visão Superficial da World Wide Web

"O navegador Web e muitos dos outros aplicativos que surgiram durante o início da comercialização da Internet e já utilizavam um protocolo cliente/servidor simples : o cliente inicia uma conexão com um servidor conhecido, transfere alguns dados por download e desconecta. Quando o usuário tiver terminado de recuperar os dados, o processo será repetido. O modelo é simples e direto. Funciona para tudo, desde navegar na Web a assistir a uma seqüência de vídeo. A máquina do cliente da Web não precisa ter um endereço permanente ou conhecido. Ela não precisa de uma conexão contínua com a Internet. Não precisa conciliar vários usuários. Só precisa saber como fazer uma pergunta e ouvir a resposta".(ORAM, 2001)

Do ponto de vista técnico, a World Wide Web corresponde a um conjunto distribuído de clientes e servidores executando um protocolo específico do nível de aplicação da arquitetura Internet — o protocolo HTTP ou HyperText Transfer Protocol.

Através deste protocolo, os clientes podem acessar informações diversas tendo como ponto de partida páginas escritas numa linguagem especialmente projetada para esta finalidade — a linguagem HTML ou HyperText Markup Language.

Um documento escrito em HTML corresponde a um documento com características de Hipermídia, onde a presença de hiperlinks (ou vínculos) é o aspecto mais importante. Por ser hipermídia, os hiperlinks podem ser associados a palavras ou sentenças, mas também a outros objetos. É comum associar hiperlinks a imagens gráficas, por exemplo, “botões”, de modo a tornar a interface mais agradável e intuitiva para o usuário. Uma outra possibilidade é, no caso de sites com imagens fotográficas, colocar versões reduzidas de fotografias e, associadas a estas colocar hiperlinks para dar acesso às mesmas imagens em tamanho original. Enfim, as possibilidades são diversas. Hiperlinks presentes numa página HTML podem apontar para:

- outras páginas HTML no mesmo servidor ou em outros servidores;
- outros recursos disponíveis, como gráficos, vídeo, sons, arquivos binários, etc.;
- outras partes da mesma página, através do conceito de “âncora”.

Clicando sobre os hiperlinks das diversas páginas Web que são exibidas num “browser”, o usuário realiza uma verdadeira “navegação” pelas informações distribuídas pelo mundo afora. Um usuário pode partir de uma página localizada no servidor de sua empresa ou de sua casa e, com poucos cliques de mouse, dar uma volta parcial pelos servidores distribuídos nas diversas partes do mundo.

4.2.2 Servidores e Clientes WWW

O ambiente WWW é baseado no conceito de cliente/servidor, com diversos servidores espalhados pelo mundo. Além disso, o WWW é um sistema dinâmico, independente de uma plataforma específica e inclui recursos multimídia. Para resumir o funcionamento, um programa "cliente" WWW (browser) roda em seu computador. Ele apresenta documentos transferidos de outro computador, caracterizado aqui como "servidor". O usuário pode requisitar uma busca, seguir uma ligação por um link de

hipertexto ou preencher formulários. A resposta do servidor (que com freqüência é uma máquina completamente diferente em alguma outra parte do mundo) é um documento que pode ser um texto plano, um hipertexto, ou que incorpore diversas mídias.

A principal idéia por trás da arquitetura cliente-servidor é estruturar o sistema como um conjunto de processos cooperativos (servidores) que oferecem serviços aos processos de usuário (clientes). No modelo cliente-servidor a comunicação entre processos se dá exclusivamente através da troca de mensagens. Isto torna o modelo adequado à implementação de sistemas distribuídos, uma vez que os mecanismos de comunicação podem ser desenhados para permitir a troca de mensagens entre processos executados em nós distintos.

Geralmente o modelo cliente-servidor faz uso de protocolos de comunicação simples do tipo requisição/resposta. A fim de obter um serviço, um cliente envia uma requisição ao servidor. Este, por sua vez, executa as operações associadas ao serviço e envia uma resposta ao cliente, contendo dados ou um código de erro caso o serviço não possa ser executado.

Esta dissociação entre programa cliente e programa servidor e o suporte de uma conexão de rede entre eles favorece diversas características muito bem vindas, tais como:

- ?? Um cliente pode interagir com diversos servidores ao mesmo tempo
- ?? Informações podem ser obtidas de servidores em qualquer parte do mundo;
- ?? Links podem apontar para qualquer coisa que possa ser apresentada, tornando o sistema multimídia;
- ?? O usuário fica livre de conhecer comandos, opções e linguagens para ter acesso a informações através da rede;
- ?? Novos protocolos e novas mídias podem ser incorporados tornando o sistema ainda mais flexível e abrangente;
- ?? Menus e diretórios podem ser convertidos para hipertextos, tornando ainda mais poderosa e amigável a ferramenta de comunicação;

?? Independente do servidor, a interface do usuário é sempre a mesma, então os usuários não precisam entender e se ajustar entre diferentes protocolos e interfaces.

4.2.2.1 Clientes WWW

Um cliente WWW é um software (ou browser) que permite "folhear" documentos hipermídia distribuídos na rede. O browser é o lado cliente na arquitetura cliente-servidor definida pelo WWW. Existem diversos servidores WWW, que, através do protocolo HTTP fornecem páginas (documentos hipermídia) escritas em HTML. A forma de visualizar estas páginas é através de browsers e eles são muitos (NCSA Mosaic, Lynx, MacWeb, WinWeb, Cello, Netscape ...). Na verdade, os browsers WWW conhecem não somente o protocolo HTTP, mas outros como Gopher e NNTP, para dar suporte a outras aplicações de rede.

O primeiro navegador a tornar-se popular para World Wide Web foi o NCSA Mosaic, o qual ainda é distribuído para diversas plataformas. Na **Figura 4.2**, é apresentada a interface deste navegador. Apesar de ter sido o software pioneiro, os browsers mais populares atualmente são o Navigator, desenvolvido pela Netscape e o Internet Explorer, distribuído pela Microsoft. As **Figuras 4.3** e **4.4** ilustram trechos das janelas destes dois navegadores.

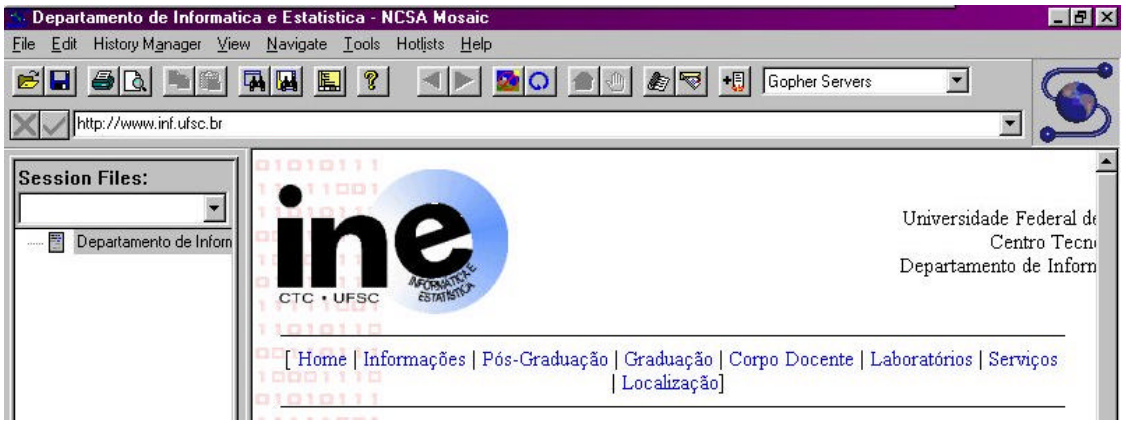


Figura 4.2 - Trecho da janela do NCSA Mosaic

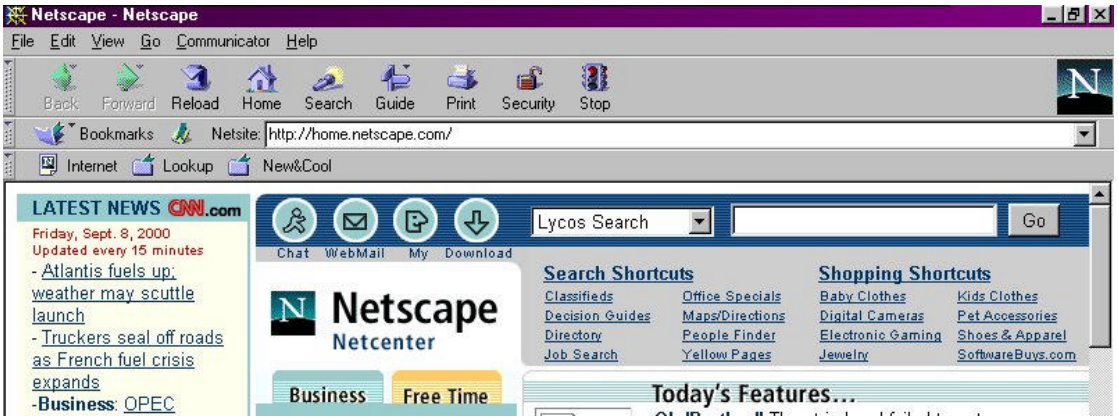


Figura 4.3 - Trecho da janela do Navigator da Netscape.

Figura 4.4 - Trecho da janela do Internet Explorer da Microsoft

4.2.2.2 Servidor WWW

Um servidor WWW é um programa que fica à espera de requisições de clientes (browsers) por documentos HTML ou informações de outros tipos (arquivos de imagens e sons, dentre outros). Também trata as requisições de realização de serviços (executados pelo servidor, na máquina do servidor) através de scripts, bem como mantém estatísticas sobre acessos. O protocolo utilizado é o HTTP e o servidor WWW tem domínio sobre a base de informações mantidas naquele site, permitindo o controle de acesso para documentos, com autorização de senhas e outras restrições. Outros servidores implementam transações seguras, com emprego de recursos de criptografia.

4.4.3 Os Endereços no Ambiente WWW (URL)

Os endereços no WWW são os URLs (Uniform Resource Locator). Os URLs são a forma padrão de designação de documentos pela qual é possível ter acesso a estes através da INTERNET. Por exemplo, de qualquer ponto do mundo é possível referenciar a homepage do servidor do INE da UFSC pelo URL http://www.inf.ufsc.br/.

Através do URL, é possível especificar a máquina (WWW), o domínio (inf.ufsc.br), o arquivo que contém o documento (ufsc/index.html), e o protocolo pelo qual ele será acessado (http). Outros formatos de URL existem para denominar outras formas de acesso a elementos da INTERNET (por exemplo, arquivos em FTP ou Gopher, artigo na Usenet ou registros de dados em uma base de dados qualquer).

4.2.3 HTML

HyperText Markup Language - HTML é uma coleção de estilos, ainda muito limitados, usados para definir os vários componentes de um documento WWW. É baseado na SGML (Standard Generalized Markup Language), a qual é usada para descrever a estrutura geral de vários tipos de documentos. A HTML preocupa-se com o conteúdo do documento e não com sua aparência ou estilo. Um documento HTML não precisa, necessariamente, ser usada com HTTP ou apresentado por um browser WWW. Pode ser usado para correio eletrônico hipertexto, sistema de news, e em qualquer lugar onde um sistema básico de hipertexto seja necessário.

Os browsers, além de prover as funções de rede para recuperar documentos dispersos, são formatadores HTML. Quando você carrega um documento HTML em um browser, ele lê (passa por um parser) as informações HTML e formata o texto e as imagens na tela de acordo com as informações contidas no documento formatado em HTML. Estes incluem vários níveis de cabeçalho, listas, menus e formatação de estilos de texto, dentre outros.

4.2.4 HTTP

HyperText Transfer Protocol - HTTP é um protocolo padronizado para transferência, através da rede, de arquivos que contém documentos hipermídia. Além de um protocolo para transferência de hipertexto, HTTP foi projetado para dar acesso à informação com eficiência necessária para realizar saltos de acordo com a exigência dos hipertextos. HTTP transfere principalmente documentos HTML, mas na verdade está aberto para suportar um ilimitado e extensível conjunto de formatos. É o protocolo mais popular entre os suportados por URLs. URLs do tipo HTTP seguem a forma básica do URL.

Dada a importância deste protocolo, a seção a seguir deste capítulo vai trazer uma visão mais detalhada sobre a operação do protocolo HTTP.

HTTP : O Protocolo de Transferência na WEB

O protocolo HTTP (HyperText Transfer Protocol) foi desenvolvido para atuar como o meio de comunicação entre clientes e servidores no ambiente World Wide Web, independente do formato de informação a ser transferida. A comunicação através do protocolo HTTP é viabilizada através do uso de soquetes (ou sockets) TCP/IP.

Os navegadores Internet são considerados clientes HTTP, enquanto os servidores Web são considerados servidores HTTP.

Recursos HTTP

Embora muitas pessoas considerem HTTP como um protocolo para transferência de arquivos (uma vez que páginas em HTML nada mais são que arquivos), é mais correto assumir que HTTP é um protocolo para transferência de “recursos”.

Recursos são qualquer objeto que pode ser localizado através de uma URL. O recurso mais comumente buscado na Web é, sem dúvida, um arquivo, mas, se considerarmos que resultados de consulta a bases de dados, documentos, etc., podem também ser “baixados” para o navegador, então, devemos assumir que efetivamente HTTP é um protocolo para transferência de recursos e não de arquivos.

Mensagens HTTP

A comunicação no protocolo HTTP segue o modelo Cliente-Servidor, como já foi dito anteriormente. Um cliente HTTP estabelece uma conexão com um servidor HTTP e encaminha uma mensagem de solicitação. Em resposta a esta solicitação, o servidor encaminha outra mensagem contendo o recurso que foi solicitado. Terminada a transferência do recurso, o servidor encerra a conexão. Por causa deste formato de comunicação, o HTTP é considerado um protocolo sem estado (ou stateless), uma vez que, ao encerrar a conexão, ele não guarda nenhuma informação sobre as condições de operação daquela conexão.

O formato das mensagens HTTP é o mesmo para as solicitações e para as respostas. Genericamente, as mensagens podem ser descritas por:

- uma linha inicial;
- zero ou mais linhas de cabeçalho;
- uma linha em branco;
- um corpo (opcional) da mensagem (por exemplo, um arquivo).

A linha inicial é composta diferentemente dependendo do tipo de mensagem, se esta é uma mensagem de solicitação ou de resposta. Uma linha de solicitação é composta basicamente de três componentes, separados por espaços:

?? o nome do método;

?? o caminho de acesso local para o recurso;

?? a versão do protocolo HTTP utilizada.

No caso de uma linha de resposta, ou linha de status, como ela é comumente chamada, os componentes também são três:

?? a versão do protocolo HTTP utilizada;

?? um código de status;

?? uma frase descritiva do código de status.

Uma linha de status conhecida da maior parte dos usuários da Web é a famosa HTTP/1.0 404 Not Found.

Os códigos de status são sempre compostos de 3 dígitos inteiros, sendo que o dígito mais significativo define a categoria do código. O quadro a seguir descreve as categorias existentes no HTTP.

Formato do Código	Categoria
1XX	mensagens de informação
2XX	indicação de sucesso na comunicação
3XX	redireção do cliente para outra URL
4XX	erro do lado do cliente
5XX	erro do lado do servidor

As linhas de cabeçalho

As linhas de cabeçalho fornecem informação sobre a solicitação ou resposta, ou sobre o objeto contido no corpo da mensagem. O formato geral de uma linha de cabeçalho é: nome do cabeçalho: valor

No protocolo HTTP existem diferentes tipos de cabeçalhos, alguns dos quais estão descritos no quadro a seguir:

Cabeçalho	Categoria
From:	indica o e-mail de quem está fazendo a solicitação
User-Agent:	indica o programa no cliente que encaminha a solicitação
Server:	indica o servidor utilizado para definir uma resposta
Last-Modified:	indica a data de última modificação do recurso acessado
Content-Type:	indica o tipo do recurso encaminhado numa resposta
Content-Length:	indica o tamanho (em bytes) do recurso

O corpo da mensagem

O corpo das mensagens segue as linhas de cabeçalho e são utilizados para conduzir os recursos solicitados para o cliente. Quando uma mensagem conduz um recurso em seu "corpo", existem linhas de cabeçalho para prestar informações sobre o recurso conduzido, como no caso dos cabeçalhos Content-Type e Content-Length.

Exemplo de troca de mensagens HTTP

Para acessar uma página localizada num dado servidor, por exemplo, em http://www.somehost.com/path/file.html, a seguinte mensagem é enviada:

```
GET /path/file.html HTTP/1.0
From: someuser@jmarshall.com
User-Agent: HTTPTool/1.0
[blank line here]
```

A resposta vinda do servidor terá o seguinte formato:

```
HTTP/1.0 200 OK
Date: Fri, 31 Dec 1999 23:59:59 GMT
Content-Type: text/html
Content-Length: 1354

<html>
<body>
<h1>Happy New Millennium!</h1>
(more file contents)
  .
  .
  .
</body>
</html>
```

Métodos HTTP

As trocas de mensagens no protocolo HTTP incorrem na ativação de métodos, onde os mais comuns são GET e POST.

O Método GET

Este é o método utilizado para localizar e obter informações disponíveis nos servidores HTTP. O formato de uma mensagem do tipo GET é similar ao formato já descrito para as mensagens, sendo que, no caso das respostas, uma linha de status é conduzida e, se o recurso solicitado foi encontrado, este é trazido no corpo da mensagem de resposta.

No caso deste método, é possível estabelecer condições para o envio ou não do recurso. Isto é feito introduzindo-se cabeçalhos específicos, tais como aqueles apresentados na tabela a seguir:

Tabela 4.1: Cabeçalhos específicos do método GET.

Cabeçalho	Categoria
`If-Modified-Since:`	verifica se o recurso foi modificado desde a data indicada
`If-Unmodified-Since:`	verifica se o recurso não foi modificado desde a data indicada
`If-Match:`	verifica a existência de determinados tags no recurso
`If-None-Match:`	verifica a ausência de determinados tags no recurso
`If-Range:`	verifica que partes de um recurso foram modificadas

O Método HEAD

O método HEAD é similar ao método GET, mas com uma diferença fundamental. Na resposta, o recurso solicitado não é trazido, mas apenas as linhas de cabeçalhos correspondentes. Este método é útil quando um cliente quer obter informações sobre um determinado recurso, mas sem fazer o download do mesmo.

O Método PUT

O método PUT pode ser visto como um oposto do método GET, uma vez que, ao invés de ir buscar recursos no servidor, ele serve para enviar recursos para o mesmo. Seu uso básico é encaminhar páginas Web para servidores HTTP, ou seja, ele é utilizado para viabilizar a "publicação" de páginas num dado servidor.

O Método POST

O método POST é utilizado basicamente para enviar dados para um servidor, dados estes que serão processados por alguma entidade presente. Um exemplo de utilização freqüente do método POST é o envio de dados postados num formulário HTML para processamento por um script CGI.

Embora o método GET possa ser utilizado também para encaminhar dados a um servidor para posterior tratamento, é interessante destacar aqui as diferenças com relação a este método:

- ?? os dados são enviados no corpo da mensagem HTTP e é feito uso de cabeçalhos específicos (como `Content-Type:` e `Content-Length:`) para prestar informações sobre os dados;
- ?? o identificador único de recurso (URI) não é um recurso para ser localizado, mas sim um programa que vai manusear os dados transmitidos;
- ?? a resposta ao método é normalmente a saída de um programa e não um arquivo estático.

O Método DELETE

O método DELETE, como já se pode imaginar, é utilizado para solicitar a eliminação do recurso indicado pela URI. No caso deste método, os atributos de autenticação e permissões desempenham um papel de importância no uso deste método, uma vez que eles podem atuar como o principal dispositivo de proteção contra a eliminação de material importante, devido a uma solicitação vinda de um usuário não autorizado.

O Método TRACE

Este método é utilizado para ativar um loop-back de aplicação da mensagem solicitada. O receptor desta pode ser o servidor original ou um proxy da rede. Métodos TRACE não incluem recursos e são utilizados comumente para fins de teste ou diagnóstico, uma vez que eles podem fornecer informações sobre o que está sendo recebido do outro lado da conexão.

4.3 A colaboração e a Internet

A Internet não modifica, sozinha, o processo de colaborar. É um meio que estimula para que a colaboração ocorra de maneira mais rápida, já que a presença física

não é obrigatória. As qualidades, estratégias e ferramentas usadas na Internet são reconhecidas por promoverem a aprendizagem colaborativa.

Para PAAS (1999), a qualidade freqüentemente reconhecida da Internet, que a estabelece como uma das mídias mais voltadas à aprendizagem colaborativa é a possibilidade de se comunicar a qualquer momento, a qualquer hora com pessoas em qualquer lugar. Outras qualidades da Internet voltadas à colaboração podem incluir:

- facilidade de uso;
- possibilidade de interagir no tempo real ou não (assíncrono/síncrono);
- possibilidade de publicar e visualizar informações;
- capacidade de servir como repositório de dados e informações ou documentos compartilhados.
- Alteração no modo em que as pessoas colaboram e interagem entre si;

Estão cada vez mais comuns organizações utilizarem a Internet como ferramenta ou ambiente para colaboração entre seus membros, visto suas vantagens e qualidades, fazendo com que esta possa ser chamada de redes colaborativas. Uma "Rede Colaborativa" é um conjunto de aplicativos e ferramentas Internet para facilitar a publicação, troca e gerenciamento de informações oficiais de um grupo de pessoas ou entidades com interesses ou alvos semelhantes. Tecnicamente ela pode ser entendida como uma espécie de Intranet/Extranet. Sempre incorporada a uma base de dados que pode ser distribuída ou centralizada, uma rede colaborativa fornece ferramentas e ambientes on-line para facilitar que todos seus membros possam contribuir com informações para a construção distribuída de um repositório do seu conhecimento.

4.3.1 Intranet/Extranet

Segundo HILLS (1997) uma Intranet é uma Internet interna, e proporciona a comunicação e colaboração em uma organização ou empresa. Uma Intranet tem a mesma aparência e formato técnicos do WWW (World Wide Web) da Internet, apresentando as informações armazenadas numa base de dados, através de hipertexto e hípermídia na rede local (LAN-Local Area Network) da organização.

A Intranet se diferencia da Internet por ser utilizada principalmente dentro da instituição. As informações circulam com maior velocidade, pois geralmente trata-se de uma rede local com maior largura de banda (bandwidth)

Da mesma forma que a Internet, uma Intranet é baseada em protocolos TCP/IP (Transmission Control Protocol/Internet Protocol) em conjunto com a Hiper Text Transmission Protocol (HTTP) e o Hiper Text Mark-up Language (HTML) que permitem a troca de informações e arquivos independente de plataforma ou tipo de computador usado. Por isso, uma Intranet/Extranet igualmente pode rodar aplicativos colaborativos (feitos em linguagens como Java, C++, etc.).

"As ferramentas de intranet ajudam tanto no desenvolvimento simultâneo como no compartilhado". (HILLS, 1997)

Uma Extranet é simplesmente a utilização da infra-estrutura de comunicações da Internet para se comunicar com outras Intranets. Para proteger a base de dados e informações da Intranet do acesso de qualquer usuário, um sistema de segurança chamada de Firewall[10] é usado. Um esquema simplificado de uma Intranet/Extranet em uma empresa é demonstrado na figura 4.5.

---

10 situa-se entre a rede interna e a parte externa à Internet. Eles filtram pacotes, escolhendo quais poderão passar e quais serão proibidos de trafegar. (ORAM, 2001)

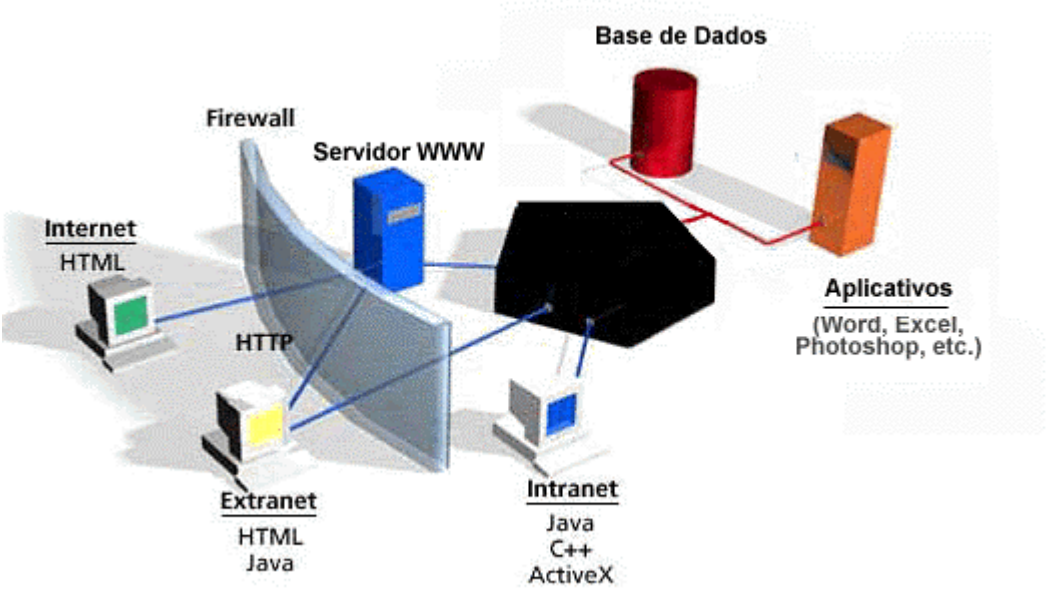


Figura 4.5 Esquema Simplificado de Intranet/Extranet (PAAS, 1999)

4.4 Benefícios do uso da Internet/Intranet

É possível destacar os seguintes benefícios no uso da Internet/Intranet como interface de ferramentas:

?? Acesso mais rápido e atualizado as informações;
?? Redução de custo ;
?? Economia de tempo;
?? Aumento na produtividade e eficiência operacional.

Considerando a colaboração os benefícios ainda são:

- Aperfeiçoamento da comunicação (velocidade, compreensão, consistência, disponibilidade, universalidade), (HILLS, 1997);
- Compartilhamento de conhecimento e colaboração;
- Aperfeiçoamento e melhoria na qualidade da atividade envolvida (tanto para empresa quanto para o cliente).

4.5 Conclusão

O gerenciamento e compartilhamento eficiente de informação em nível da organização estão sendo cada vez mais enfatizados por ferramentas de indexação, busca e recuperação de informação entre departamentos, aplicativos de compartilhamento e edição de documentos (document sharing and editing) , e sistemas internos de e-mail.

A facilidade de uso, o alcance mundial e o custo relativamente baixo da Internet/Intranet, que são baseadas na simplicidade dos browsers e protocolo TCP/IP, faz com que o WWW seja uma interface ideal para ferramentas que proporcionem a colaboração síncrona ou assíncrona. Há ainda navegadores Web para todos os sistemas operacionais, o que elimina a necessidade de criar e adaptar diferentes aplicações para cada plataforma.

Os navegadores, ou browsers tendem a se tornar uma interface universal com o usuário, portanto os profissionais empregarão bastante tempo acessando informações e comunicando-se por meio deles. A Web está rapidamente tornando-se a plataforma de computação estratégica para o futuro.

Com isto as empresas que produzem software proprietário cliente/servidor e groupware estão aderindo e habilitando suas aplicações para a Web. Como resultado, as aplicações proprietárias estão se tornando mais abertas e poderão ser acessadas por qualquer navegador com a segurança apropriada. Ainda, os navegadores Web são fáceis de usar e quase não requerem treinamento, concentrando em como usar as aplicações específicas para obter benefícios nos negócios.

Com a globalização a competitividade mudou. Os concorrentes estão espalhados pelo mundo, e não somente do outro lado da rua. Desta forma, os clientes podem ser globais também. Tudo isso é possível graças à tecnologia, e a Internet/Intranet/Extranet contribuem facilitando a comunicação e colaboração já que a força de trabalho (os desenvolvedores) não precisa estar concentrada em um único local.

Capítulo 5

Ferramentas de Controle de Versões

Desenvolver software em equipe é algo cada vez mais comum nas empresas, pois a distância não se constitui mais como uma barreira, visto que a Internet é uma tecnologia a qual dá o suporte para esta afirmação. O desenvolvimento de software é um processo consumidor de tempo e de recursos financeiros além de necessitar a experiência em diversas áreas do conhecimento. Assim, alguns projetos são realizados por diversas equipes que trabalham concorrentemente.

Com o crescimento do uso da informática juntamente com o desenvolvimento crescente de tecnologia, os sistemas têm-se tornado cada vez maiores. Então é necessária uma ferramenta para controlar e gerenciar este desenvolvimento? Se a organização ou empresa possui membros de sua equipe de desenvolvimento em locais remotos ou não, existem grandes vantagens quando estas são adotadas, algumas tais como a redução com locomoção, compartilhamento de informações, reutilização de código já que versões anteriores são armazenadas, e ainda comparação entre as mudanças efetuadas entre dois arquivos. A não utilização destas ferramentas pode gerar mais trabalho, visto que o controle sobre as mudanças, tais como quem fez, quando, e, uma descrição do realizado é de extrema importância.

O propósito deste capítulo é descrever algumas das principais opções de softwares e ferramentas de uso privado ou aberto existentes hoje, específicos para o controle de versão de um projeto ao qual uma organização pode adotar para que se possa propor um modelo adequado ao controle de versão tendo a Web como interface.

Para uma melhor visão das ferramentas comerciais disponíveis, este capítulo descreverá uma análise e comparação entre algumas selecionadas. As ferramentas estão citadas em ordem alfabética, sendo utilizado a Internet como fonte de pesquisa, bem como contato com alguns colaboradores que se utilizam ou já se utilizaram alguma citada.

Produtos verificados

5.1 - Code Co-op - Version Control System

O Code Co-op acessado em 27/08/02, (RELIABLE SOFTWARE, 2002) do site http://www.relisoft.com/co_op é uma ferramenta que permite o controle de versões de componentes em desenvolvimento de forma transparente em relação a localização geográfica dos desenvolvedores, sem a necessidade de um servidor central, armazenando os objetos de forma distribuída entre os desenvolvedores. A administração do projeto é feita através de um administrador de usuários que estão envolvidos, com privilégio de acesso em 4 níveis.

Para o uso desta ferramenta é preciso que a equipe tenha conhecimento técnico de desenvolvimento em grupo. Possui sincronismo de serviços de e-mail ou rede local para avisar a equipe quando é realizado o check-in de um arquivo.

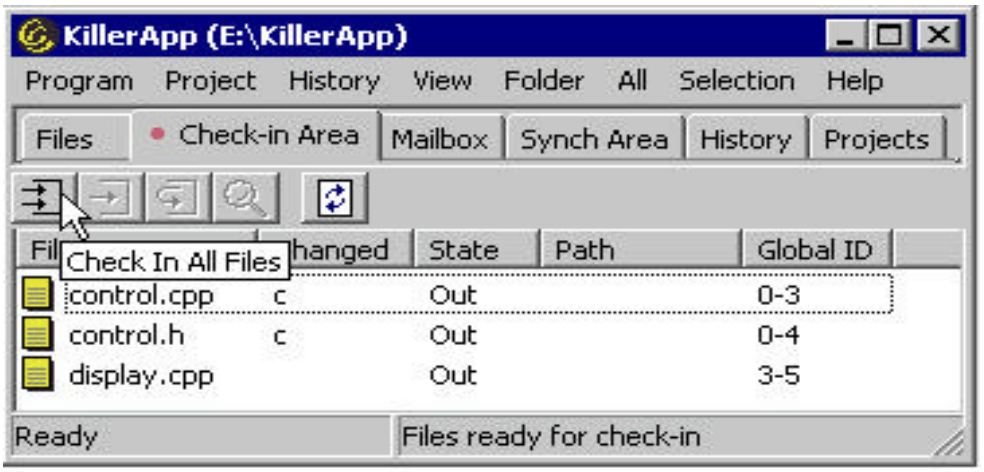


Figura 5.1 . Interface do Code Co-op quando efetuado Check-in em um arquivo selecionado

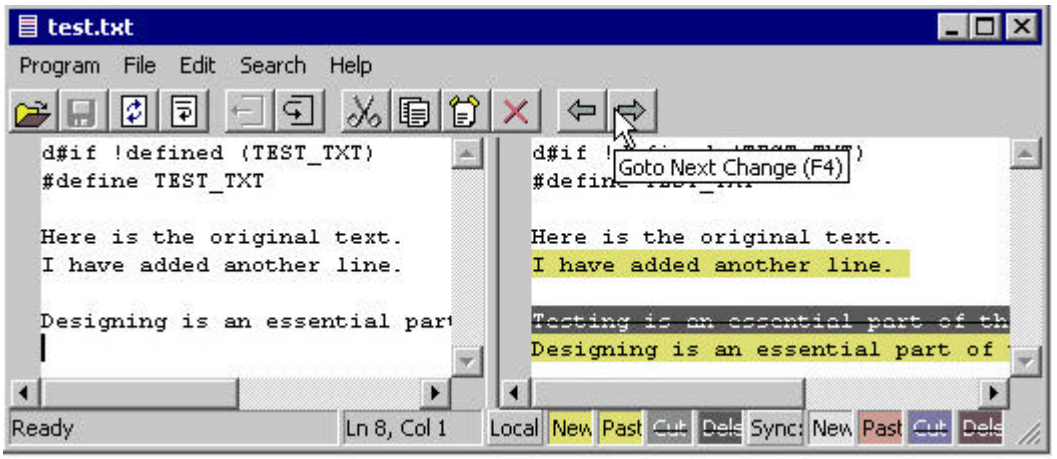


Figura 5.2 Visualização do arquivo test.txt criado a partir da função de diferenças . O lado direito mostra as mudanças que tinham sido feitas e o lado esquerdo mostra o estado final do arquivo.

Através do VCS - Version Control System todas as funções padrões tais como check-in[11] (Figura 5.1), check-out[12], abrir, editar, restaurar e visualizar histórico e comparar diferenças entre dois arquivos (Figura 5.2) estão disponíveis.

5.2 - CVS - Concurrent Versioning System

Em http://www.cvshome.org, acessado em 28/06/2002, a ferramenta CVS - *Concurrent Versioning System*, ou sistema concorrente de versões permite o controle de um projeto. Pode ser utilizada livremente para desenvolvimento de sistemas do tipo open-source (código aberto - Mozzila, GIMP, GNOME), onde vários colaboradores são capazes de operar simultaneamente o mesmo projeto em plataformas diferentes (Figura 5.3), sendo possível também ser usado individualmente. (ARNSON,2002)

O código fonte pode ficar em um repositório na máquina local, quando for apenas um colaborador, ou em repositório no servidor quando se tratar de uma equipe de colaboradores.

O acesso dos usuários pode ser efetuado através do sistema operacional em uso ou através do próprio CVS, onde é possível criar duas listas especificando qual a permissão

[11] Check in é verificação quando da devolução do arquivo alterado ou não em um dado local.
[12] Check out é a verificação de retirada de um arquivo, se há permissão para isto.

do usuário (leitura ou gravação) Quando o colaborador solicita através de um Check-out uma cópia da última versão, estas listas são verificadas e caso o usuário esteja nas duas a prioridade será somente de leitura do arquivo. Se as modificações estiverem prontas, é possível efetuar um Check-in, inserindo-as novamente no repositório do CVS. A cópia na máquina do usuário (colaborador) ficará apenas read-only, ou seja, permissão somente de leitura. Com o recurso *history* é possível verificar o histórico das alterações do arquivo selecionado e também recuperar alguma versão anterior, não sendo necessário ser o administrador do projeto.

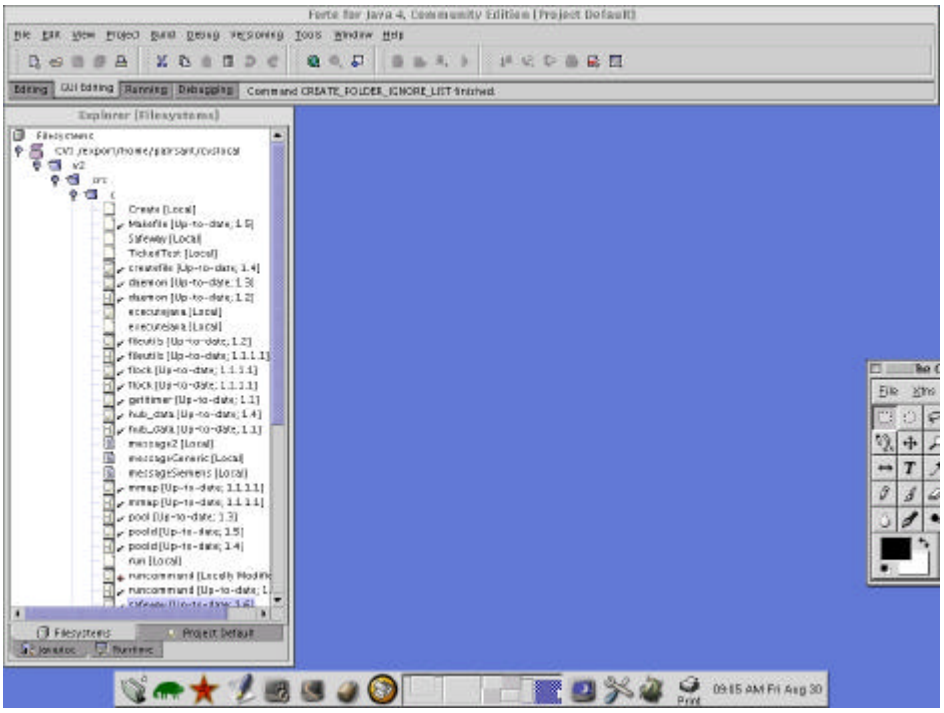


Figura 5.3 . Visualização da interface CVS em ambiente Forte da SUN.

Segundo BLANDY (2002), o CVS pode responder as seguintes questões

- Quem fez a alteração?
- Quando foi realizada ?
- Por que foi realizada ?

O controle do código na versão tradicional é efetuado quando é realizado o Check-out do arquivo e quando modificado no Check-in de volta. O colaborador que dá Check-out do arquivo tem direito exclusivo para modificá-lo. Neste momento, nenhum outro

colaborador pode fazer Check-out , a não ser o administrador do sistema, até que seja efetuado o Check-in do arquivo, mas o CVS suporta desenvolvimento paralelo, permitindo que mais de uma pessoa trabalhe em um mesmo arquivo ao mesmo tempo.

Outro recurso do CVS é quando você tentar enviar um arquivo para o repositório, ele verifica se a versão que você possui é a mais recente. Se duas pessoas simultaneamente fizerem alterações a diferentes partes do mesmo arquivo, o CVS é suficientemente e fundi as alterações. Mas se duas pessoas fizerem alterações à *mesma* parte do arquivo, o CVS não consegue determinar qual deve ser o resultado final, por isso gera um conflito . Isto normalmente ocorre quando o colaborador submete uma alteração e um segundo, sem correr o CVS update para receber as alterações do primeiro, tenta submeter as suas mudanças.

5.3 – PVCS – Sistema de Controle de Versões

O PVCS – Version Manager da INTERSOLV – http://www.merant.com/products/pvcs/vm/index.asp - acessado em 22/08/2002 (MERANT), é um sistema que permite gerenciar e controlar toda a etapa de desenvolvimento de um projeto, desde a documentação obtida através do processo de análise ao controle dos arquivos com os códigos fontes e seus executáveis. É direcionado tanto para equipes de pequeno a grande porte, oferecendo a elas o recurso de desenvolvimento, administração e coordenação dos códigos fonte (Figura 5.4). Cada desenvolvedor tem o software cliente instalado em sua máquina, podendo também optar pelo acesso remoto ao servidor (Version Manager Server), através de um browser.

Esta ferramenta se utiliza do locking - mecanismo que o PVCS usa para prevenir que outros usuários possam alterar o mesmo objeto. Quando um colaborador, utilizando a versão não remota, solicita determinado componente para fazer uma operação de update ou correção é realizada uma cópia para sua máquina (diretório público – Figura 5.5) (Check-out). Caso outro desenvolvedor solicite para manutenção

um componente que se encontra bloqueado, precisará aguardar pela liberação do mesmo pelo desenvolvedor que bloqueou o componente (Check-in).

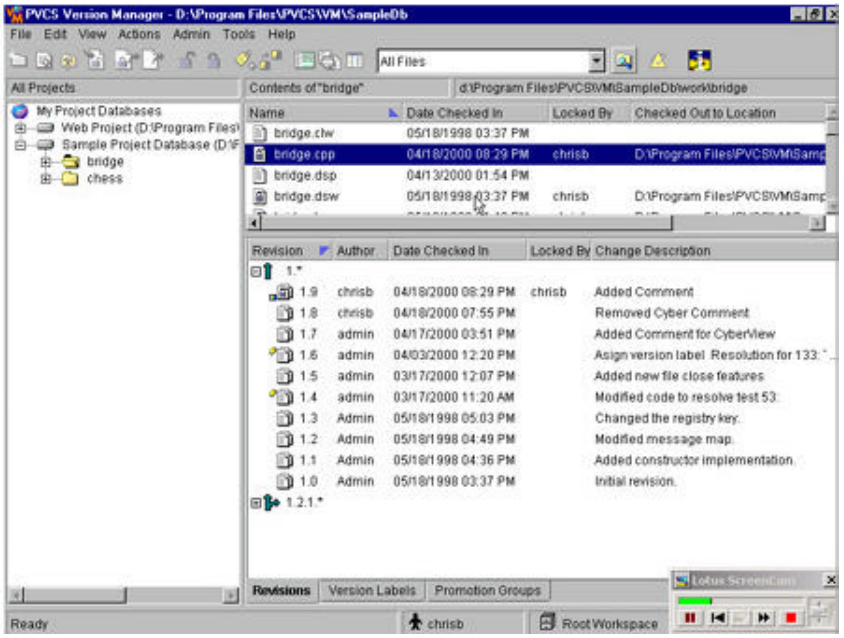


Figura 5.4 Representação hierárquica sobre as versões de um sistema PVCS Version Manager – interface não baseada na Web

O controle de versões processa-se automaticamente, sendo gravado a data e hora que este ocorreu, usuário, comentários, além das alterações detectadas nos componentes no momento em que é realizado o Check-in. Além disso, há uma ferramenta responsável pelo monitoramento dos projetos, que avisa todos os desenvolvedores envolvidos sobre as alterações efetuadas (PVCS Pulse).

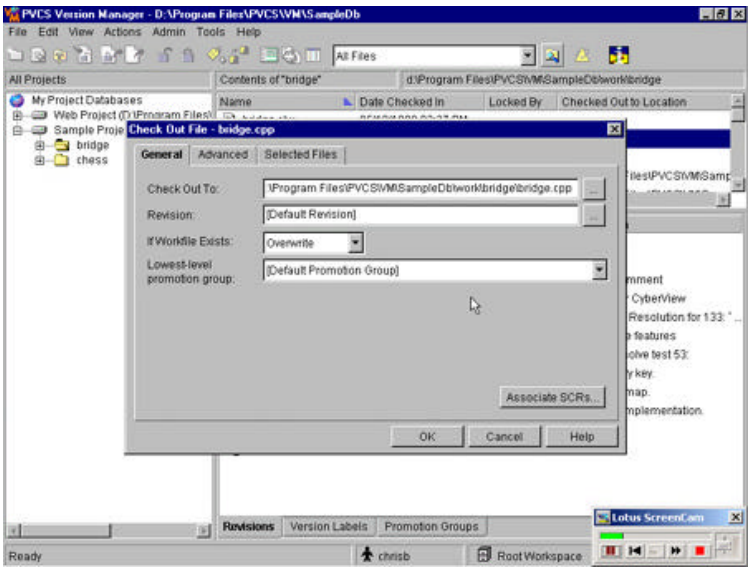


Figura 5.5. Imagem que mostra o momento que é efetuado Check-out de um arquivo . PVCS Version Manager

Esta oferece ainda uma *feature*[13] que a partir da seleção de arquivos em duas ou mais revisões, exibe a diferença (linhas de código) entre eles (*Merge Tool*-Figura 5.6). Além da comparação e verificação de diferença nas linhas de códigos é possível, através de outra *feature* visualizar as diferenças obtidas nos testes das versões realizados anteriormente (Figura 5.7).

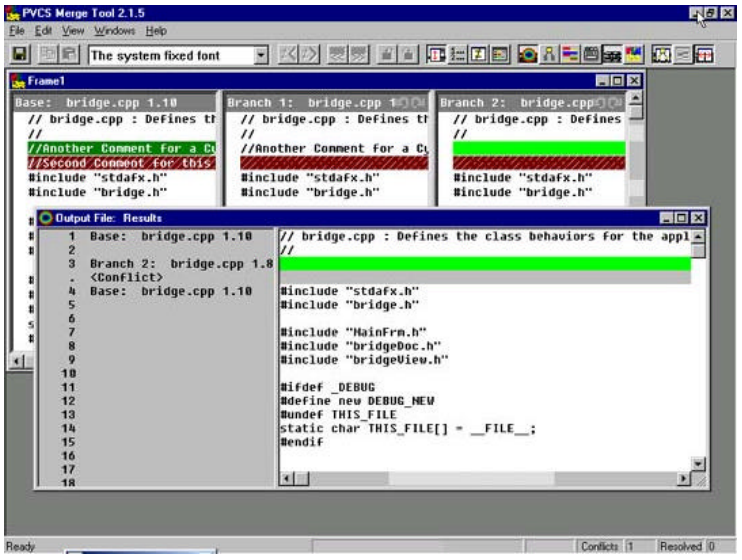


Figura 5.6 Tela identificando as diferenças entre três arquivos de revisão selecionados. PVCS Merge Tool Version Manager

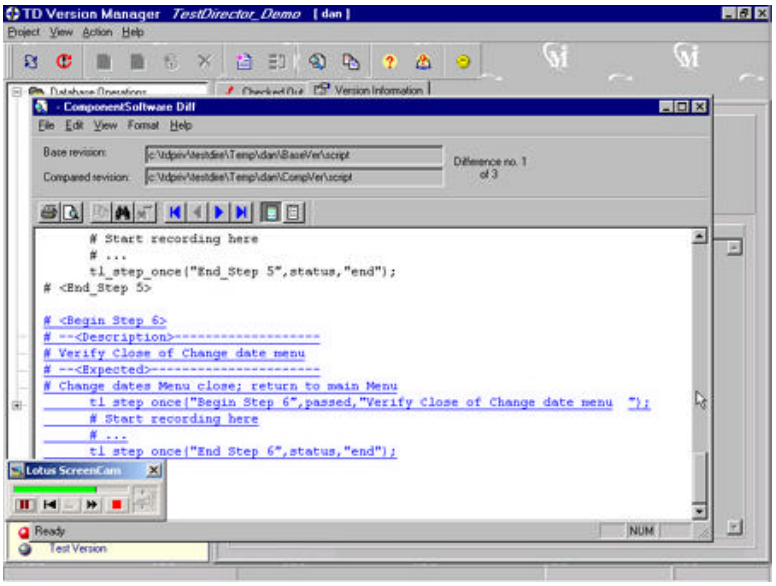


Figura 5.7 TestDirector Version Manager – comparação entre testes das versões

O sistema PVCS – *Version Manager*, permite também um novo nome a cada versão do projeto, podendo associar a este números e codinomes.

5.4 – Rational ClearCase

O Rational ClearCase, http://www.rational.com , acessado em 27/09/2002 (RATIONAL, 2002) é um software para o Gerenciamento de Configuração (SCM) que oferece um completo controle para pequenas equipes localizadas em um mesmo local (figura 5.8) ou grandes equipes distribuídas geograficamente (Figura5.9) durante os ciclos de desenvolvimento de software, assegurando a recuperação de qualquer versão e organizando um processo efetivo de desenvolvimento distribuído.

Esta ferramenta possibilita o desenvolvimento paralelo. O controle dos arquivos segue o padrão de check In e Check out. Utiliza o mecanismo de UCM - Unified Change Management , que oferece recursos no desenvolvimento de equipe desde o início até a versão final,ajudando a reduzir riscos na coordenação do projeto e priorizando atividades de desenvolvimento.

---

[13] função especial, ou capacidade, ou, projeto de hardware ou software;

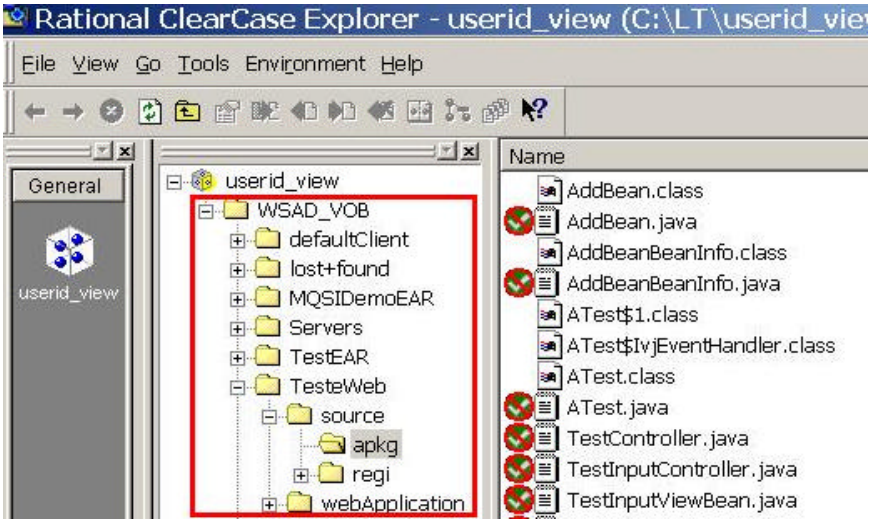


Figura 5 8. Interface Explorer

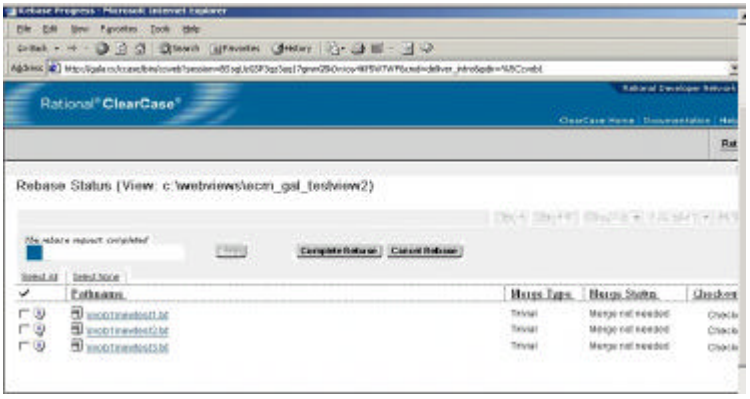


Figura 5.9 Interface Browser

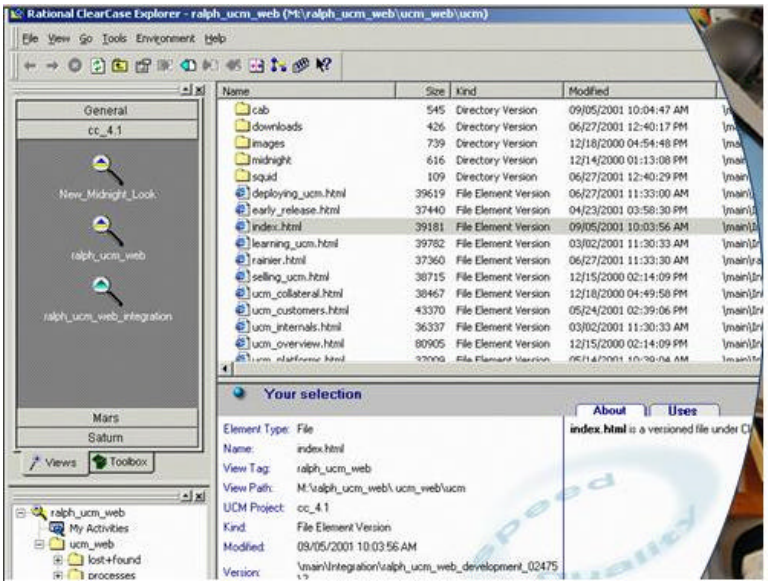


Figura 5.10 Localização dos arquivos

5.5 - RCS – Revision Control System

O RCS - Revision Control System, http://www.csc.calpoly.edu/~dbutler/tutorials/winter96/rcs/, acessado em 27/09/2002 (RCS, 2002), é uma ferramenta de auxílio ao controle de versões. É baseado em um conjunto de comandos UNIX, e foi desenvolvido por Walter Tichy, da Universidade de Purdue, em 1985, como um melhoramento do SCCS - Source Code Control System. Sua principal função é administrar grupos de revisões de arquivos, através do controle do acesso a eles. Para se modificar um documento sob controle do RCS, é preciso antes recuperá-lo do sistema; o RCS então gerencia esses acessos e armazena as sucessivas revisões de forma ordenada. Como as atualizações são mantidas independentes do arquivo original, pode-se facilmente retornar a revisões anteriores caso seja necessário. Uma grande vantagem do RCS é que ele não armazena integralmente todas as versões de um mesmo documento, mas apenas as diferenças existentes entre as sucessivas versões. Isso implica em grande economia de espaço de armazenamento. O bloqueio do arquivo quando é retirado para alteração é feito manualmente, caso o desenvolvedor esqueça de fazê-lo outro poderá alterá-lo também.

5.6 - Team Coherence Version Manager

O software Team Coherence Version Manager http://www.qsc.co.uk/gpv/gpversion.htm, acessado em 29/08/2002 (QUALITY SOFTWARE, 2002)é um controlador de versões para códigos escritos em Delphi , C++ Buider e Visual Basic para uso de grandes equipes de desenvolvedores.

Utiliza-se de um repositório central de componentes (Figura 5.11), o que faz com que possa ser utilizado por equipes distribuídas. O acesso aos arquivos deste repositório é efetuado a partir do nível de permissão que o usuário tenha. Para fazer as modificações é necessário fazer o Check-out destes arquivos (Figura 5.12), que ficam marcados como estando sendo usados exclusivamente. Até que seja efetuado o Check-in o arquivo fica apenas liberado como de leitura.

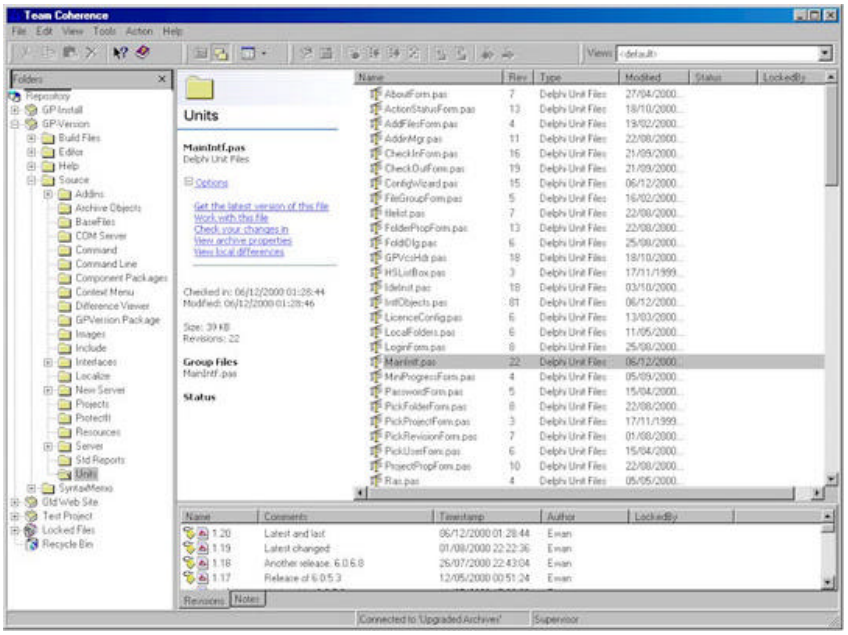


Figura 5.11 Tela principal do Team Coherence Version Manager com as informações do projeto que estão divididas em janelas que descrevem os detalhes.

Figura 5.12 Visão da função de Check out

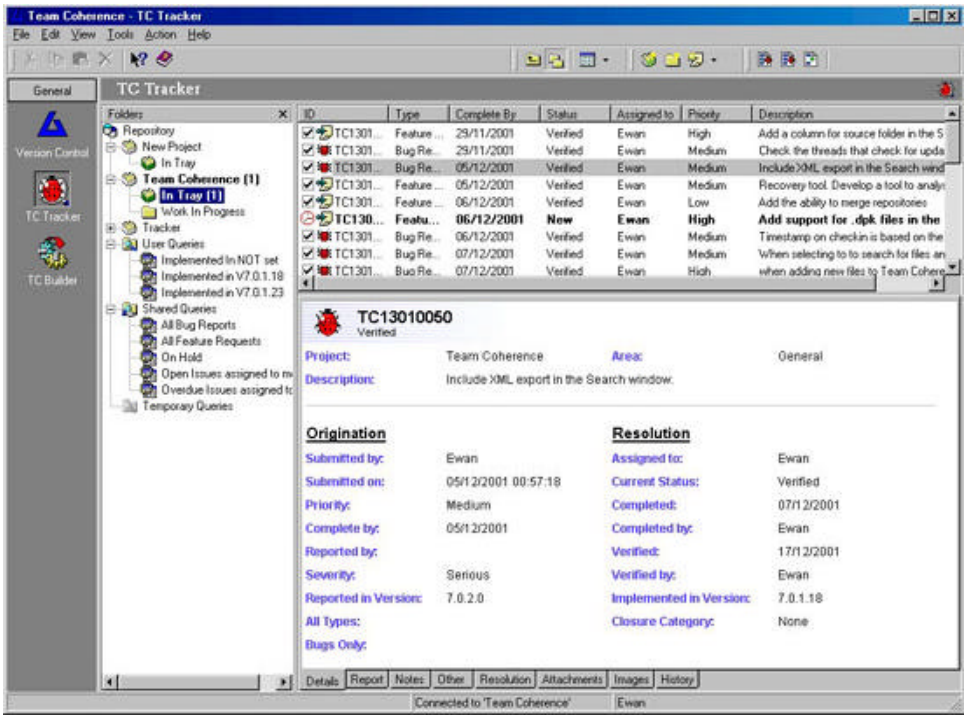


Figura 5.13 Módulo principal do TC Tracker integrado com o Team Coherence. Os campos configurados em 'HTML' tem links para editar e modificar apenas o requerido

5.7 - TeamStudio CIAO!

O TeamStudio CIAO!! , acessado em 25/07/2002 (CIAO, 2002) http://www.gaia.inf.br/web%5Cgaiahome.nsf/TeamStudioCiao?OpenPage , é um sistema de controle de entrada, saída e controle de versão para ambiente Lótus Notes e Domino[14]. Identifica qualquer alteração que tenha sido feita em um Design[15], e que vários desenvolvedores trabalhem em um mesmo Design de uma base de dados. Este guarda todas as cópias das versões anteriores juntamente com uma descrição da alteração para que possam ser restaurados (Figura 5.14) em outra base de dados.

[14] Servidor que controla o acesso ao design da base de dados vigiadas do Notes.
[15] Mesmo que Arquivo

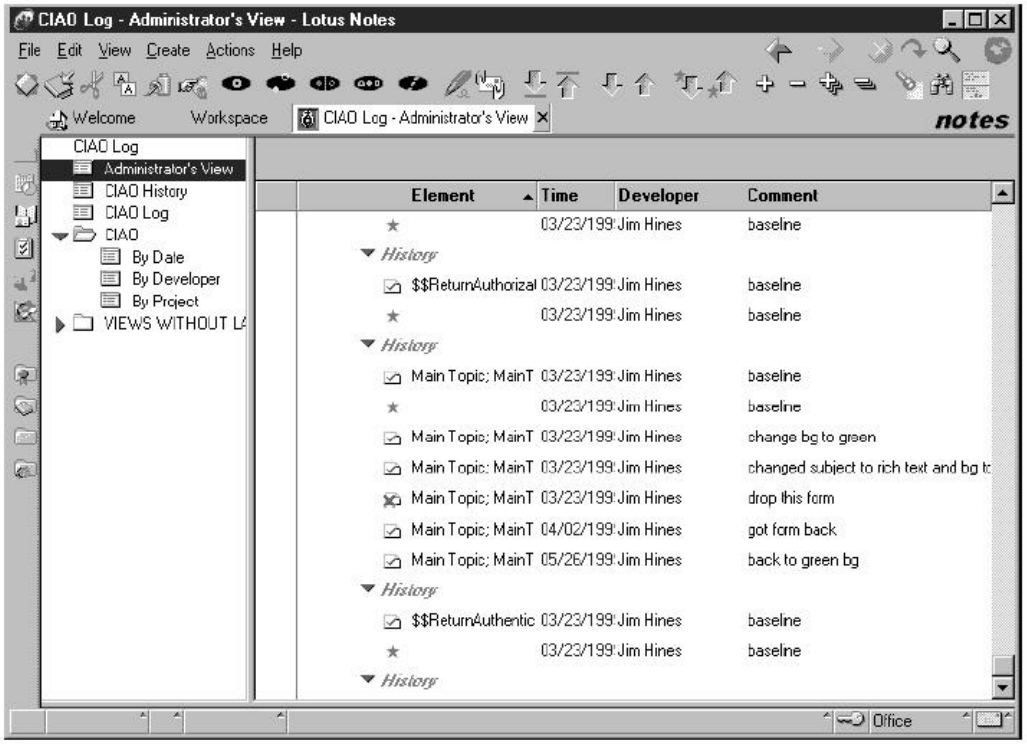


Figura 5.14. Interface do Team Studio CIAO! em ambiente Lótus Notes

Versões anteriores dos elementos de Design podem ser identificados e restaurados usando o comando Obter (Get). Quaisquer elementos de Design que tenham sido atualizados, podem ser restaurados utilizando a opção Recuperar (Recover).

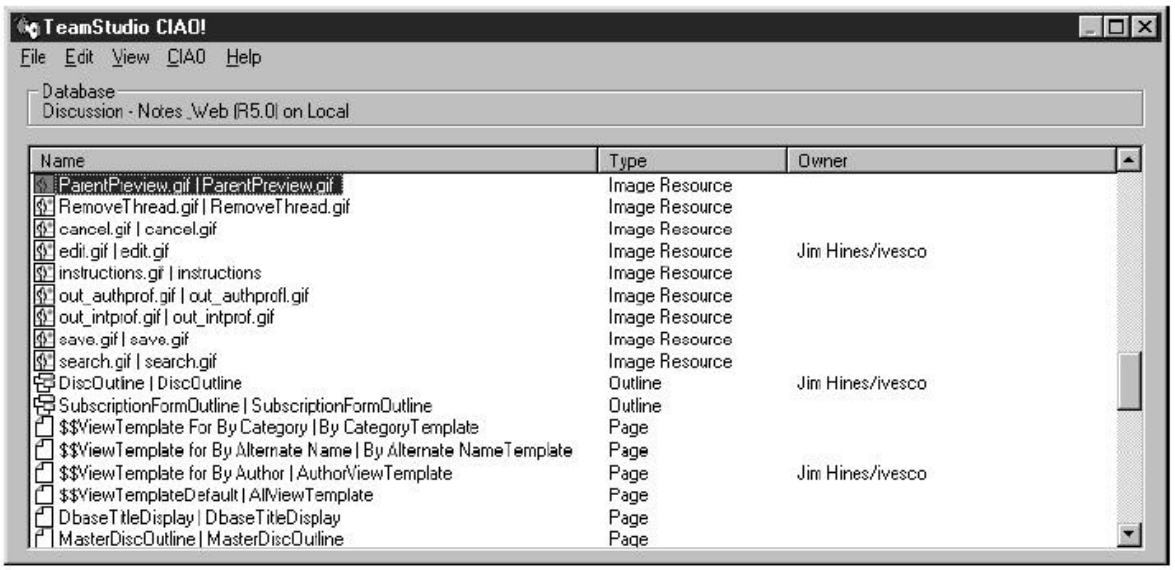


Figura 5.15 Visualizador de Referências

A CIAO! Server Edition somente protege o Design contra atualizações que não são feitas pelo desenvolvedor (Figura 5.15) que tem o elemento reservado. Não permite que os desenvolvedores controlem a entrada e a saída, nem inclui a facilidade de visualizar o histórico do Design. Para isto, é necessário instalar CIAO! Client Edition no computador de cada desenvolvedor.

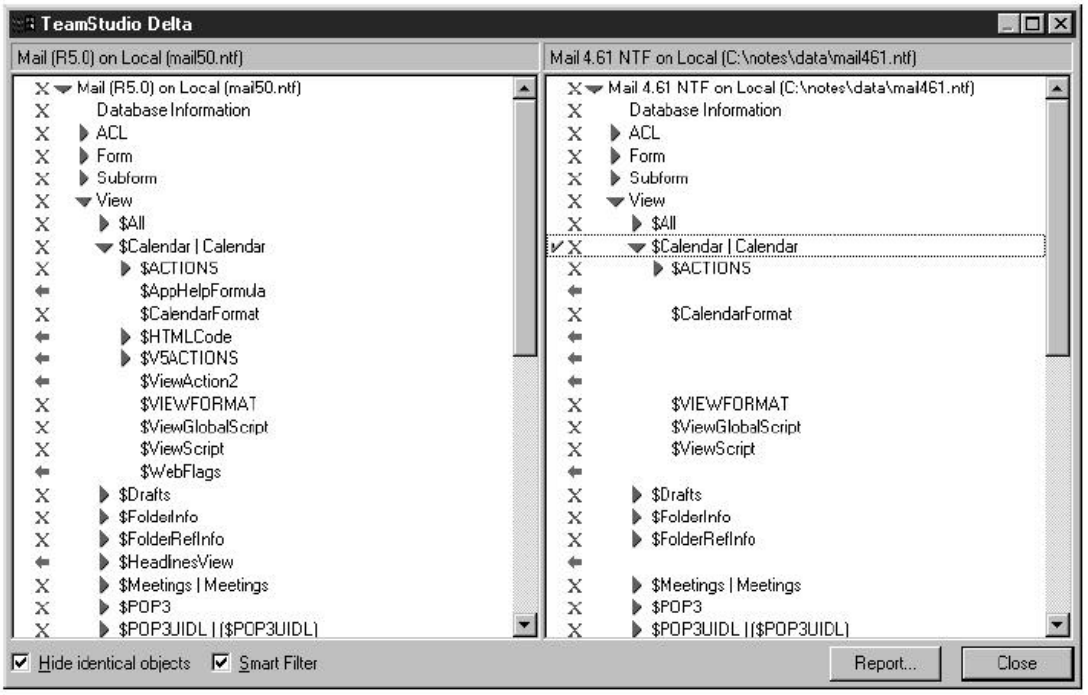


Figura 5.16. Visualização da análise comparativa entre dois Design – Team Studio Delta.

Acrescentando outra ferramenta, o TeamStudio Delta, é possível verificar uma análise comparativa entre dois Design na mesma base ou em bases diferentes (Figura 5.16), podendo gerar um relatório de diferenças.

5.8 -Tlib ™ Version Control

A ferramenta TLIB , http://www.burtonsys.com/, acessado em 29/08/2002 é uma ferramenta de uso rápido por equipes de bastante conhecimento técnico em desenvolvimento em grupo (Figura 5.17).

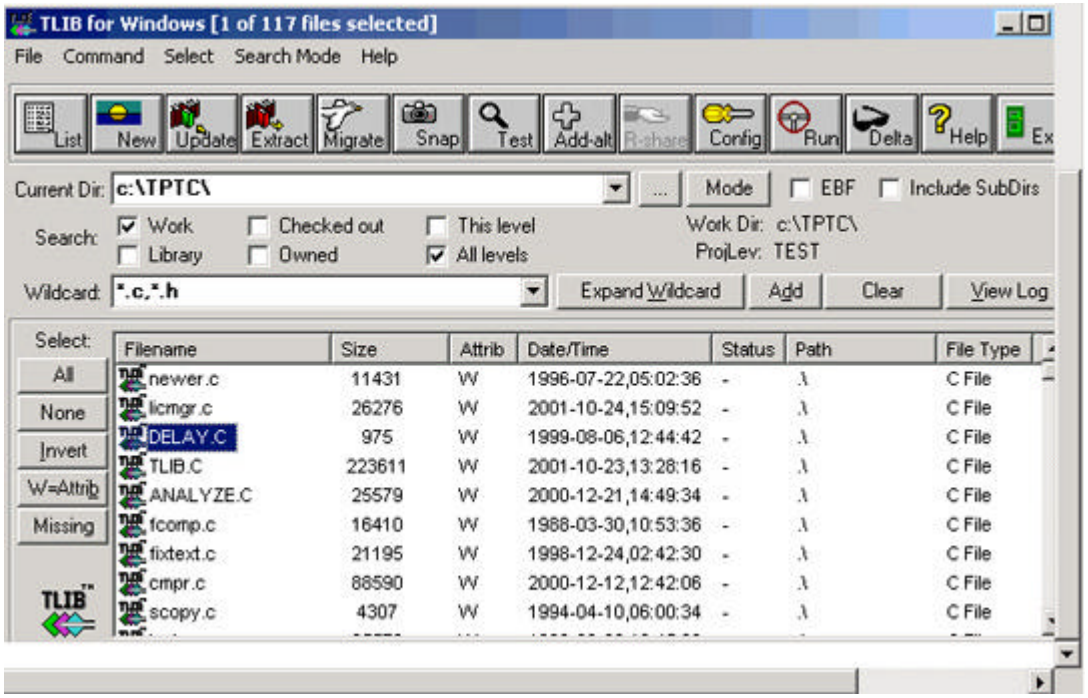


Figura 5.17 Interface do histórico sobre o arquivo selecionado. TLIB Version Control for Windows

Foi projetado com uma estrutura de armazenamento aberta, a qual permite a importação e conversão automática de outros controladores de versão tais como : PVCS, SourceSafe, MS Delta, Sorcerer's Apprentice, e outras ferramentas do RCS (MKS, GNU, Unix). Oferece ainda a integração com várias linguagens de programação

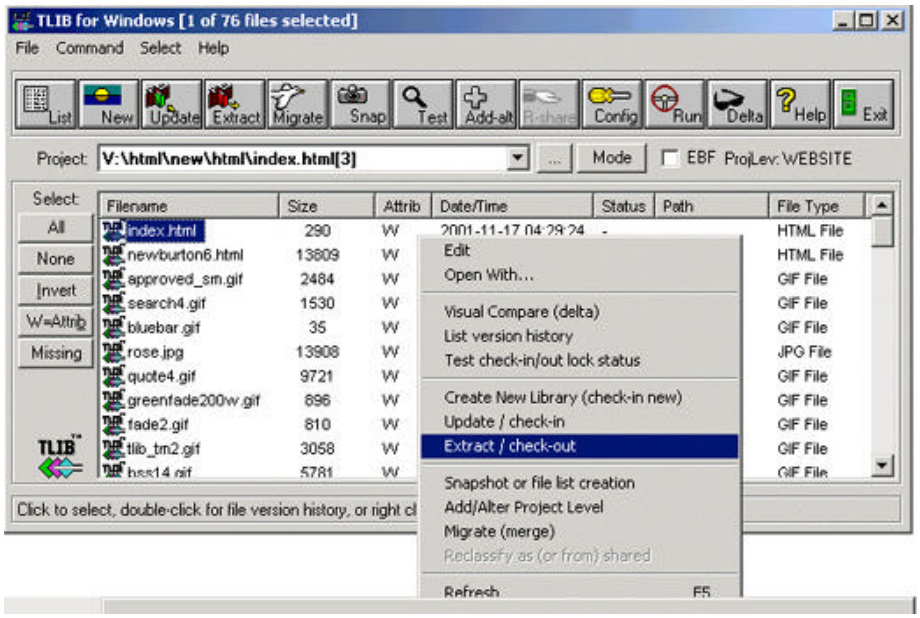


Figura 5.18 . Visualização da escolha do Check-out no TLIB Version Control for Windows

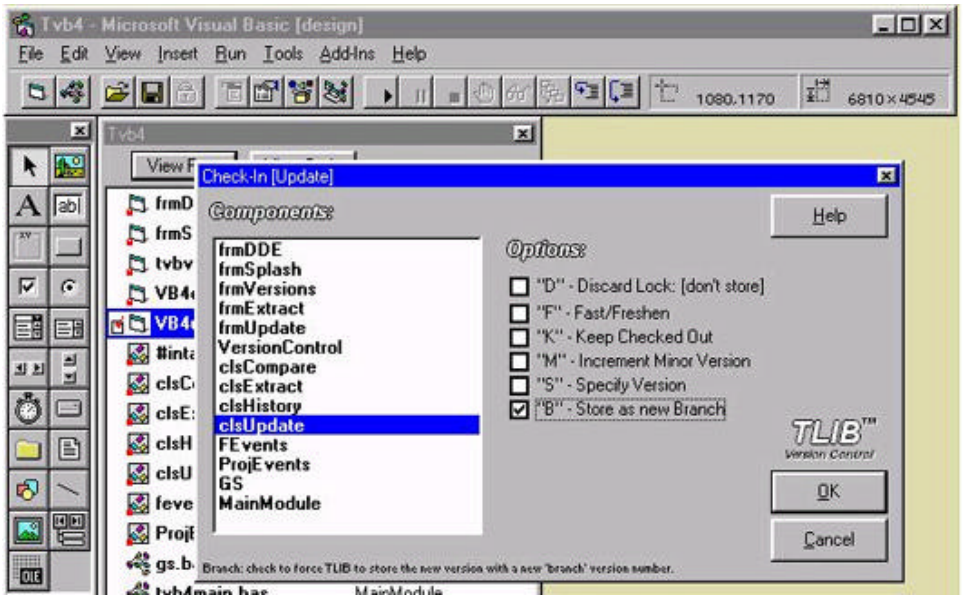


Figura 5.19 . Visualização da escolha do Check-in .Integração do Visual Basic com TLIB Version Control for Windows.

Tem todas as funções padrões tais como check-out (Figura 5.18) ,check-in (Figura 5.19) utilizando o locking para segurar que um arquivo esteja apenas sendo modificado por um colaborador em um determinado tempo. Suporta a junção (merge) simultânea de diferentes versões do mesmo arquivo de código (Figura 5.20) ou apenas entre a versão corrente e o seu antecessor.

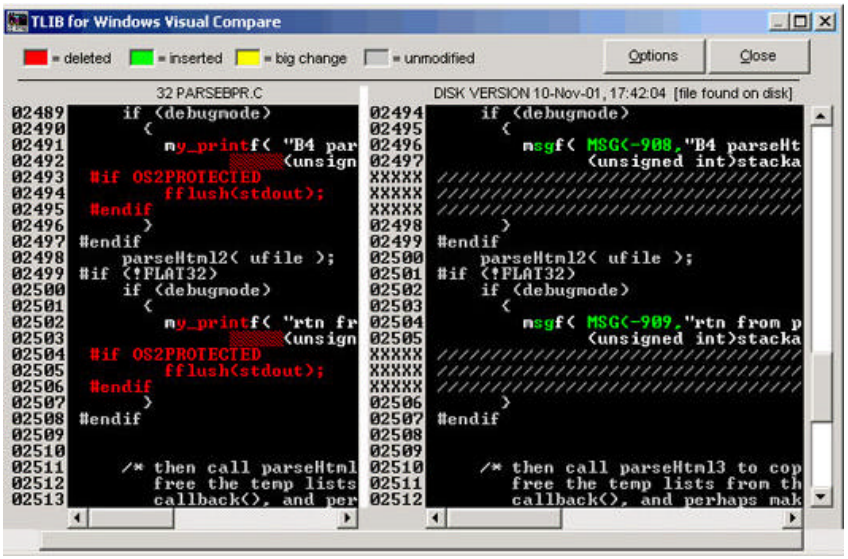


Figura 5.20 Visualização no VisualCompare das diferenças entre dois arquivos . O lado esquerdo mostra as mudanças que tinham sido feitas e o lado direito mostra o estado final do arquivo.

5.9 - VCS Lite - Version Control System Lite

O VCS Lite - Version Control System Lite da Acacia Systems é uma ferramenta de controle de versão utilizada para marcar os arquivos com descrições das versões já criadas (Figura 5.21). Este software não pode ser comparado com ferramentas mais robustas e profissionais de controle de versão, mas ela cumpre aquilo que se propõe perfeitamente e é gratuita , site http://www.acaciacons.com.au/vcslite/ acessado em 23/07/2002.

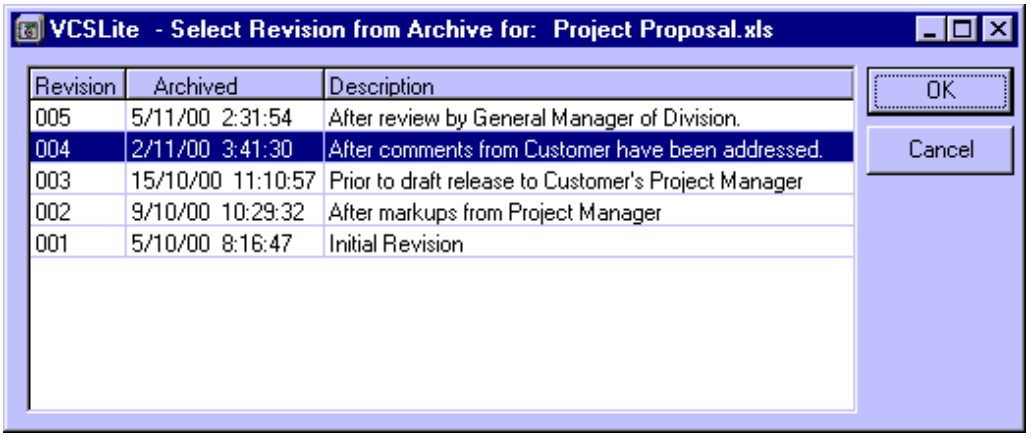


Figura 5.21 Interface do VCS-Lite mostrando as revisões de um arquivo selecionado.

Sua interface opera através do Windows Explorer utilizando o drag'n'drop ou o Right-Click & Send-To (Figura 5.22). Pode-se utilizar o VSCLite para salvar o histórico das alterações do documento, planilhas, relatórios ou arquivo de qualquer tipo para apenas um desenvolvedor, não sendo possível utilizá-lo por uma equipe de desenvolvimento concorrente.

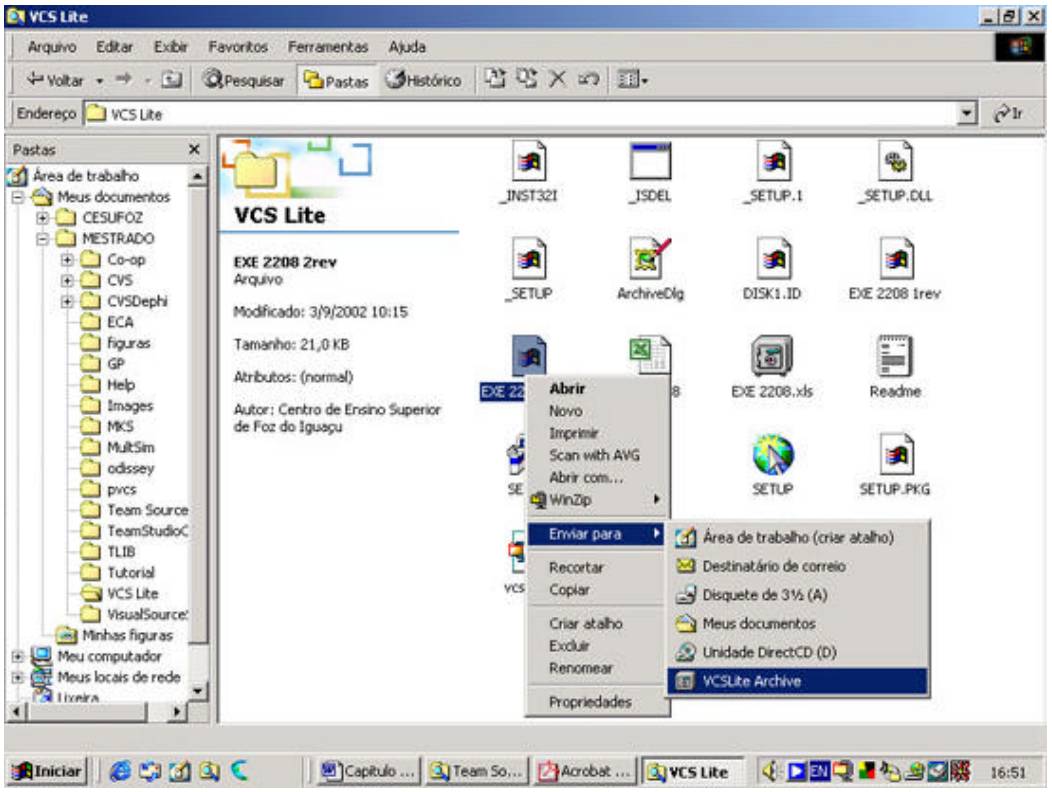


Figura 5.22 Visualização da interface quando é realizada a seleção do arquivo para controle da versão.

O arquivo de dados anterior ao modificado é armazenado em um arquivo com a adição da extensão *.vca* . Por exemplo: se o arquivo a modificar é myfile.txt, será identificado como *myfile.txt.vca* , sendo sempre armazenado em diretório diferente do diretório de trabalho.

Como é de uso individual não oferece recursos de bloqueio do uso de arquivos, e tampouco comparações de diferenças entre versões.

5.10 - Visual SourceSafe

O Visual SourceSafe , da MICROSOFT , como pode ser verificado em http://msdn.microsoft.com/ssafe, acessado em 22/06/2002, é um gerenciador e controlador de desenvolvimento de projetos (Figura 5.23) em um ambiente

extremamente flexível que se adapta a qualquer desenvolvimento, incluindo o projeto, o desenvolvimento, testes e eliminação de erros somente para plataforma Windows.

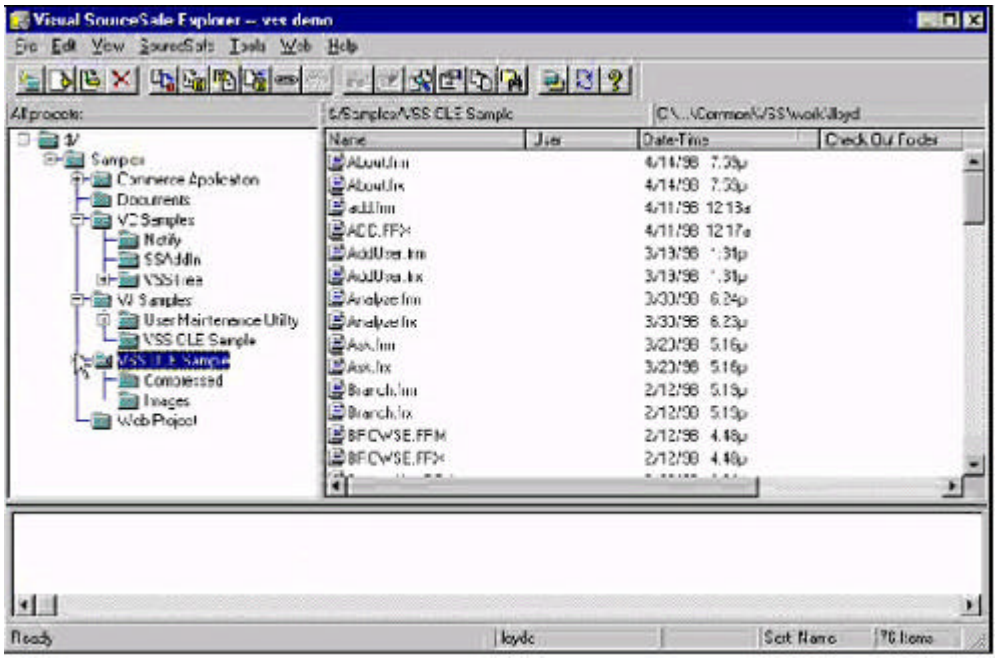


Figura 5.23 . Representação hierárquica do projeto. Microsoft Visula SourceSafe Explorer

Pode ser utilizado por equipes que tenham pouca experiência em controle de desenvolvimento colaborativo. Possibilita que cada desenvolvedor tenha uma visão total do projeto, e ainda a visualização do histórico (Figura 5.24) das modificações dos arquivos (data, hora, usuário, comentário), sendo possível recuperar versões precedentes, não ficando documentado no sistema.

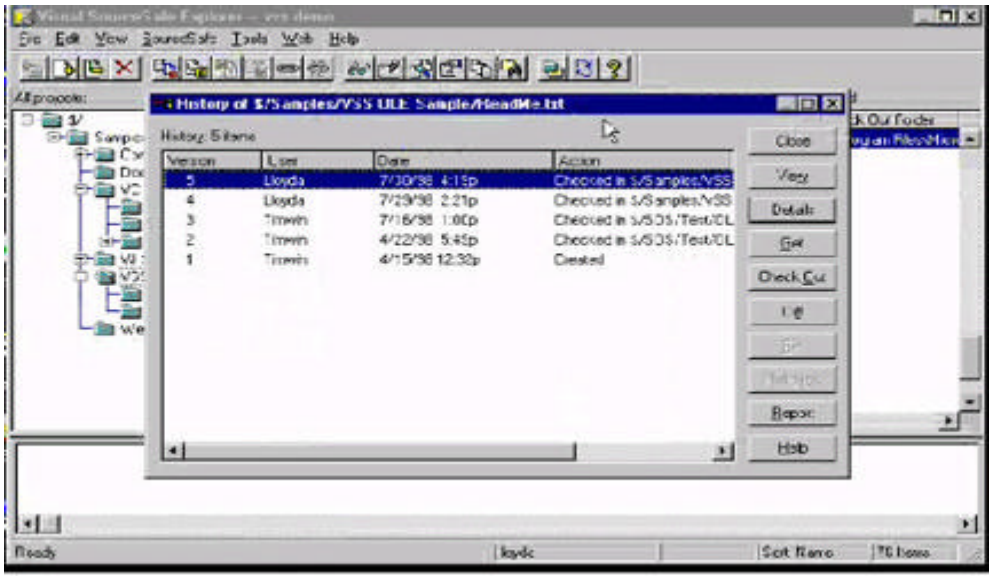


Figura 5.24 Histórico das modificações efetuadas . Microsoft Visual SourceSafe 6.0.

O Visual SourceSafe gerencia as versões dos arquivos através de um controle de acesso de usuários (5 níveis ao total), fazendo uma cópia do arquivo solicitado - Check-Out - na máquina local do colaborador que o fez, não no código original. Assim o controle da versão de código passa a ser realizada com maior segurança. Ao fazer a devolução - Check-In - uma outra cópia será armazenada no repositório do sistema (original), podendo ser acessada por qualquer colaborador que tenha permissão para tal. Desta forma, este armazena todas as cópias anteriores.

Este sistema ainda oferece ao desenvolvedor um recurso de verificação imediata de quais linhas de códigos sofreram alterações, entre as versões dos arquivos que interessam (Figura 5.25).

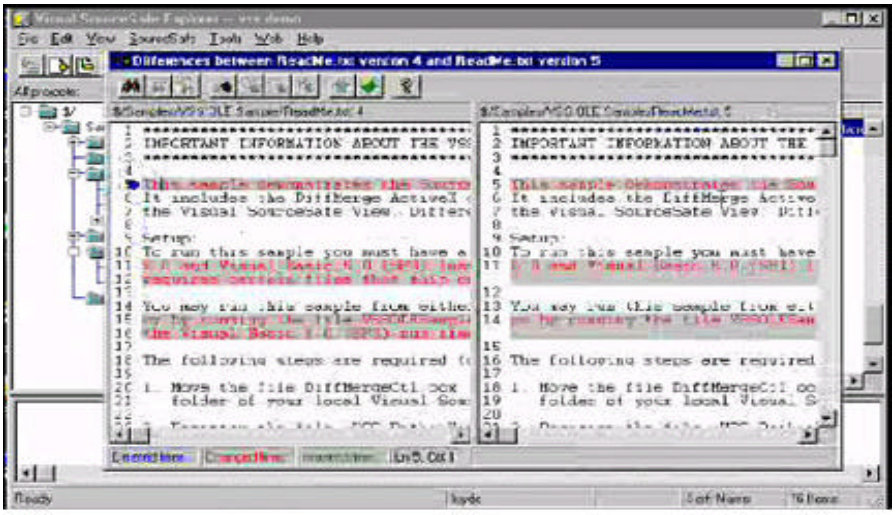


Figura 5.25 Diferenças entre duas revisões no Microsoft Visual SourceSafe 6.0. A esquerda é visualizado o arquivo mais antigo e a direita o mais atual.

5.11- Síntese Comparativa das ferramentas abordadas

	Code -Co	*CVS*	*PVCS*	*Rational Clear Case*	*RCS*	*Team Coherence*	*Team Studio CIAO*	*TLib*	*VCS Lite*	*Visual SourceSafe*
Área de Trabalho										
Versão com Interface Web	*N*	*N*	*S*	*S*	*N*	*N*	*N*	*N*	*N*	*N*
Portabilidade	*N*	*S*	*S*	*S*	*N*	*N*	*N*	*S*	*N*	*S*
Gerenciamento e armazenamento de arquivos										
Bloqueio automático do arquivo quando efetuado Check-out (Locking)	*N*	*N*	*S*	*S*	*N*	*S*	*N*	*S*	*N*	*S*
Outras ferramentas										
Visualização de Análise comparativa entre dois arquivos	*N*	*N*	*S*	*S*		*N*	*S*	*S*	*N*	*S*
Comunicação com os membros da equipe	*S*	*N*	*S*	*S*		*N*	*N*	*S*	*N*	*S*

Quadro 5.1 Comparação entre ferramentas

Em relação à portabilidade, as ferramentas que apresentam tal ponto possuem versões para várias plataformas, sendo necessário adquirir produto conforme o especificado.

Encontramos duas ferramentas de uso free,ou seja gratuito, o CVS e o VCS Lite – sendo que este último deixa a desejar no ponto principal desta abordagem que é o trabalho colaborativo.

Notou-se que para a utilização da maioria das ferramentas é preciso que a equipe tenha bastante conhecimento técnico e a integração entre o controle de versões com o ambiente de desenvolvimento, no que diz respeito a linguagem e sistema operacional utilizado, sendo isto possível mediante a aquisição de versões pré-definidas.

As ferramentas analisadas tratam os dados em nível de arquivos, identificando o colaborador, data, hora e uma descrição da alteração efetuada. Quanto à visualização de diferenças entre arquivos alterados a maioria disponibiliza este recurso sendo que o padrão é a escolha de dois arquivos, onde algumas oferecem relatórios gerados a partir destas comparações.

5.12 – Conclusão

O trabalho concorrente de equipes geograficamente distribuídas dificulta o controle de alterações nos componentes de um projeto em desenvolvimento. (PRESSMANN,1997). As ferramentas apresentadas vêm apoiar este cenário de desenvolvimento de projetos.

Todos os sistemas estudados possuem características similares de ferramentas de gerenciamento de configuração, especificamente no item controle de versões de projeto. A maneira de abordar cada aspecto é que as quais os diferenciam como observado na tabela comparativa (Tabela 5.1).

Conforme análise destes produtos é possível propor um modelo de sistema de controle de versões que agregue todos os pontos fortes encontrados proporcionando benefícios aos seus usuários em relação ao desenvolvimento em equipe, o qual será abordado no próximo capítulo.

Capítulo 6

Modelando uma Ferramenta de Controle de Versões baseada em Interface WWW

Um sistema colaborativo promove o processo de compartilhar conhecimento interativamente entre membros, com objetivo de melhoria contínua. Isto não exclui a atenção à capacidade de comunicação do membro, as habilidades do grupo, e o fornecimento do ambiente adequado para a colaboração.

A dificuldade de reunir os especialistas necessários em um mesmo local físico e a delegação do desenvolvimento de determinados componentes para outras empresas, são exemplos de fatores que podem exigir que as equipes participantes de um projeto estejam geograficamente distribuídas.

A importância de uma ferramenta para gerenciar o desenvolvimento de projetos de sistemas neste caso é realmente interessante e se faz necessária para a qualidade do produto final. Utilizando a Internet como interface, pode-se propiciar uma nova forma de colaboração entre seus membros, já que outras formas de mídias também podem ser integradas.

6.1 Considerações para modelar um sistema que integra a abordagem colaborativa à tecnologia Internet para o desenvolvimento de sistemas

Um modelo de sistema colaborativo para o controle de versões envolve o entendimento de grupos e como as pessoas se comportam nestes grupos. Também envolve um bom entendimento de tecnologias de rede e como os aspectos desta tecnologia afetam a experiência do usuário.

No entanto, muitos aspectos de grupo requerem considerações especiais. Por exemplo, grupos de milhões de pessoas não apenas se comportam diferentemente de grupos de 5 pessoas, mas também os parâmetros de desempenho das tecnologias para suportar os dois tipos de grupos são bastante diferentes. A facilidade de uso precisa ser maior para sistema colaborativo (*groupware)* do que para sistemas de um usuário porque o ritmo de uso de uma aplicação é normalmente direcionada pelo ritmo de uma interação. A capacidade de resposta e confiabilidade do sistema se torna tópicos mais significativos. É preciso ter um entendimento do grau de homogeneidade dos usuários, dos papéis possíveis que as pessoas desempenham no trabalho colaborativo.

Interface

Quando pessoas trabalham juntas com a mesma informação, elas podem individualmente desejar visões personalizadas. Neste caso do controle de versões, em ambiente distribuído , a interface deve ser idêntica para todos os usuários. Não deve haver visões personalizadas já que o projeto é o mesmo para todos, e as informações são compartilhadas . É claro que a manutenção da consistência dos dados é imprescindível, tendo controle de acesso e bloqueio de arquivos.

A hierarquia dos arquivos do projeto deve ser visível através de uma árvore de diretórios, que permita visualizar o local (em um servidor central) onde estarão armazenados, em um repositório de dados, todos os códigos do sistema no qual o grupo de desenvolvedores estará trabalhando e controlando.

A Web será a ferramenta de interface a ser utilizada, visto que é o meio no qual o controle para o desenvolvimento distribuído é o mais apropriado. Hoje o baixo custo para se conectar a uma rede viabiliza o modelo, já que grande parte das empresas já tem esta disponível. Membros da equipe de desenvolvimento pode estar em locais remotos bem como seu cliente, já que quando da manutenção ou revisão de uma versão, esta pode ser efetuada também via browser, não necessitando a presença física do desenvolvedor ou do suporte do sistema.

Portanto a Internet (Web), permite com que o desenvolvimento seja totalmente distribuído, e independente de plataforma, tanto do lado do cliente como no lado do servidor, proporciona uma interface amigável baseada em formulários HTML.

"Se você pode resolver o problema de comunicação, então os desenvolvedores podem estar em qualquer lugar." (RADDING, 1999)

Arquivos do projeto

Existe uma pressão contínua para o compartilhamento de mais informações por diversas razões. Quanto mais as informações são compartilhadas, mais facilmente será criada uma base comum de conhecimento. (INFORMATIVO, 2000).

Desta forma, armazenar e gerenciar as informações sobre cada arquivo alterado, tais como histórico de manutenções, desenvolvedor, data e hora dentre outras que sejam relevantes aos desenvolvedores, ou a equipe, é fator relevante.

É preciso prover um gerenciamento de manutenção dos arquivos, ou seja, bloquear os que estejam sendo alterados por um desenvolvedor, não permitindo assim a alteração de um mesmo componente por dois desenvolvedores ao mesmo tempo, evitando assim a inconsistência de dados.

Gerenciar as versões dos arquivos e do sistema como um todo, eliminando perdas de código e permitindo retornar às versões anteriores, mantendo também um histórico, com suas versões e as modificações efetuadas em cada versão deve ser avaliado e consideradas.

Definir políticas de manutenção, controlando os acessos aos arquivos com níveis de permissões aos desenvolvedores também é um mecanismo de segurança e privacidade para os colaboradores de uma equipe.

6.2 Arquitetura

Um Sistema Colaborativo deve ter várias características em sua arquitetura. RIZZO (2002) destaca três:

1) **Robustez** - poder replicar objetos do banco de dados , podendo armazená-los em diferentes tipos de documentos (páginas web, escritórios documentos e mensagens de e-mail) e ainda suportar replicações de servidor para servidor e servidor para cliente. É necessária esta replicação quando os usuários estão geograficamente em locais diferentes e precisam acessar diversas informações.

2) **Utilizar a Internet e a tecnologia padrão** - novas tecnologias estão cada vez mais conectadas em rede formando uma rede coesa. Um sistema colaborativo deve ser capaz de se comunicar com estas redes pela Internet, e utilizar padrões abertos para um grande número de sistemas externos como garantia da integridade de dados.

3) **Facilidade** - facilidade para utilizar tecnologia e ferramentas de desenvolvimento. O desenvolvimento deve ser aberto e os desenvolvedores podem usar diferentes ferramentas para desenvolver soluções e os usuários poderão acessá-lo também, observando suas permissões quando houver.

Conforme arquiteturas avaliadas no capítulo 2, bem como as características ora mencionadas, o modelo proposto é utilizar uma arquitetura híbrida para o ambiente do controle de versões. Este modelo teria um servidor com sua base de dados do projeto em desenvolvimento ou disponível para a equipe por completo, e que seria compartilhada. Este deverá estar na mesma máquina em que estão localizados os servidores de Web (servidor HTTP) por razões de desempenho. Certamente,poderia estar em uma outra máquina diferente daquela onde reside o servidor de Web. Porém, seria mais uma conexão a ser estabelecida para a execução dos pedidos do usuário,que teria que localizá-lo nessa outra máquina, enviar os pedidos do usuário e depois de executá-los,

teria que devolver o resultado quem o chamou na primeira máquina para que o resultado fosse enviado ao usuário.

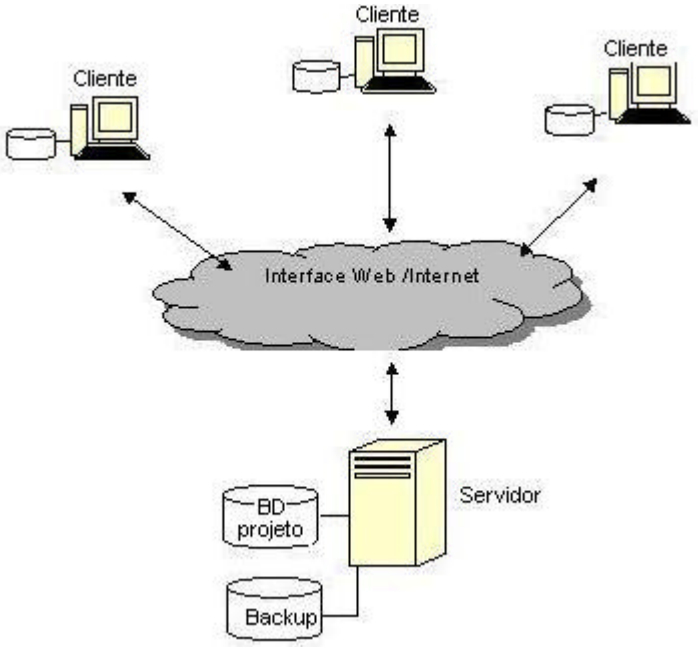


Figura 6.1 Ambiente de um sistema colaborativo utilizando a Interface Web

O compartilhamento dos arquivos é efetuado da seguinte maneira : o colaborador seleciona o arquivo de interesse ao servidor, este verifica se algum outro colaborador já não o fez. Em caso afirmativo é retornada uma mensagem, caso contrário efetuado uma cópia do arquivo solicitado para máquina local do colaborador. Assim preserva-se o arquivo fonte solicitado, ficando este apenas como leitura para os outros colaboradores até que seja efetuada a devolução.

A inconsistência não acontece pelo fato do bloqueio do arquivo quando é efetuada a solicitação (check-out), já que para liberação (desbloqueio) é preciso que o mesmo colaborador efetue a devolução (check-in). A comunicação e armazenamento são realizados pelo servidor central, e os dados divididos entre os colaboradores. O diagrama apresentado na Figura 6.1 representa este ambiente híbrido ora descrito com a interface Web (conexão http – via browser).

6.3 Modelo do gerenciamento

A atividade principal do controle de versão é o gerenciamento do projeto, quanto aos arquivos fonte. Assim, é preciso uma solução adequada para o processo, considerando os usuários, arquivos e colaboradores. Para que um arquivo não seja utilizado por mais de um desenvolvedor ao mesmo tempo, as operações acerca do arquivo serão de bloqueio – lock/(unlock)- e as atualizações ou modificações com Check-in/Check-out.

Na Figura 6.2 é possível visualizar as atividades nas quais o administrador, o colaborador e o usuário têm permissões definidas, o qual é verificado na sua autenticação.

?? **Administrador:** pessoa responsável por gerenciar todos os usuários da ferramenta e o projeto;

?? **Colaborador**: pessoas autorizadas a manipularem os arquivos que estão sob o controle de versão e que residem no repositório do servidor - desenvolvedores;

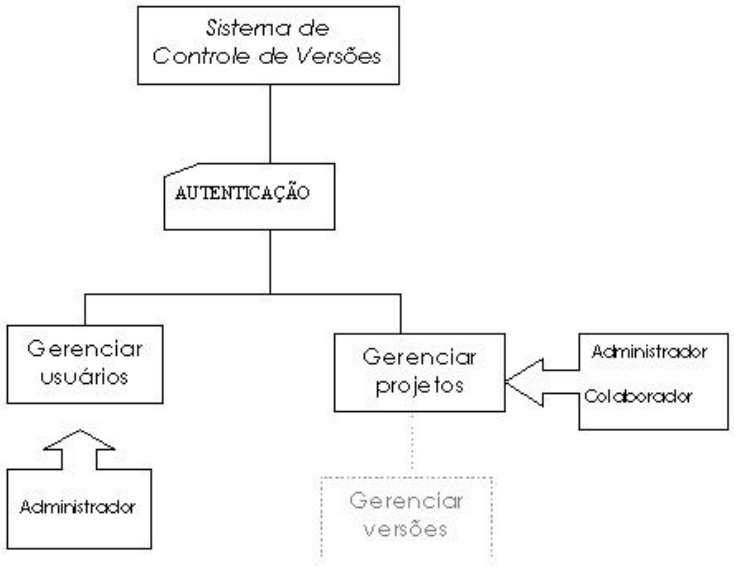
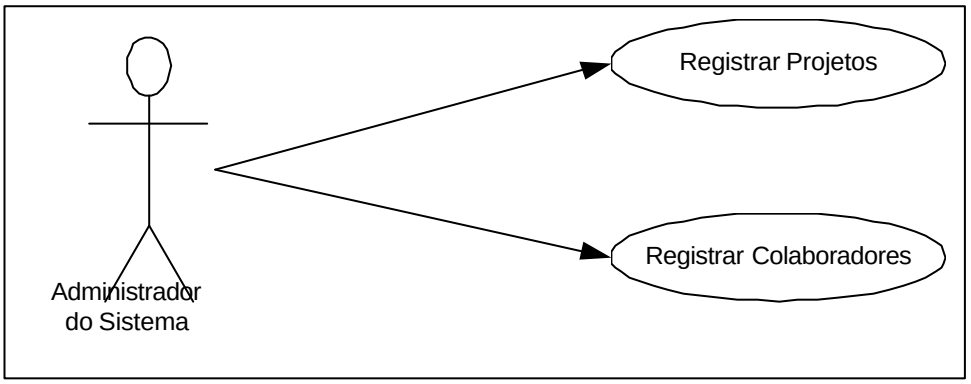


Figura 6.2 Gerenciamento das atividades do Sistema de Controle de Versões

Gerenciar projeto/colaborador

Cada colaborador é cadastrado com nível de permissão, pois pode ocorrer de algum colaborador ter permissão apenas para efetuar alterações em parte dos arquivos. Nestes dois casos o administrador, que é o responsável pelo projeto é o que tem permissão para realizar estas atividades (Figura 6.3 a Figura 6.7). O administrador, assim como os colaboradores, poderá estar em sua própria máquina (locais ou clientes) em qualquer lugar do mundo, interagindo através da interface Web (baseada em formulários HTML) e submetendo pedidos a um programa que residirá fisicamente na mesma máquina que o servidor de Web; esses pedidos, geralmente, são comandos a serem executados tais como check-out, check-in, etc.;



Caso de Uso	Registrar Colaboradores
Atores	Administrador
Finalidade	Gerenciar as informações dos Colaboradores com cada Projeto
Visão Geral	Este caso de uso é inicializado pelo ADMINISTRADOR. Ele provê a funcionalidade de cadastrar, excluir e modificar os dados de um Colaborador.
Tipo	Secundário
Referência Cruzada	Registrar Projetos

Figura 6.3 Diagrama de Use-Case do Gerenciamento de Usuários

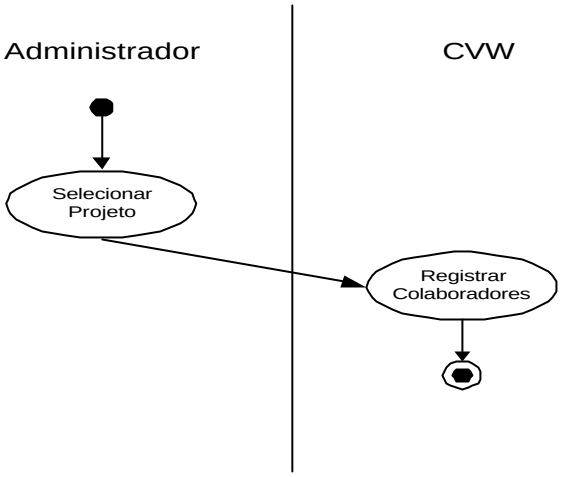


Figura 6.4 Diagrama de Atividades do Gerenciamento de Usuários

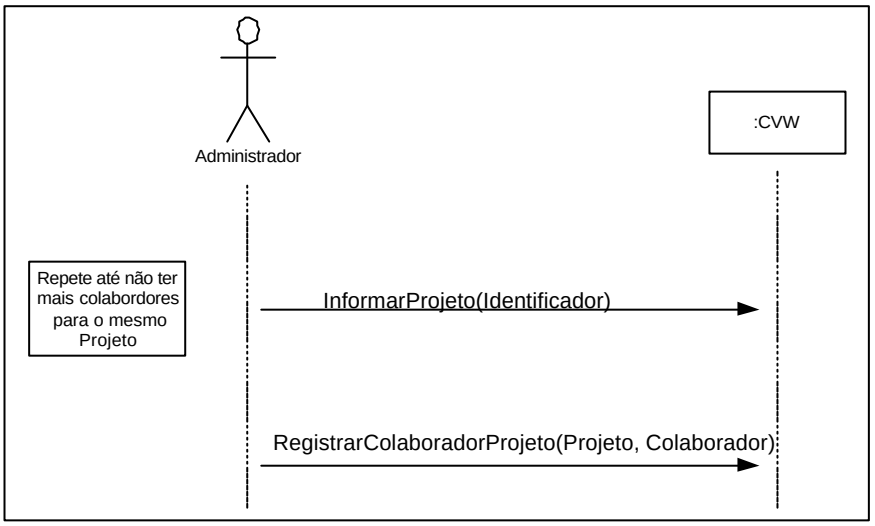


Figura 6.5 Diagrama de Seqüência do Gerenciamento de Usuários

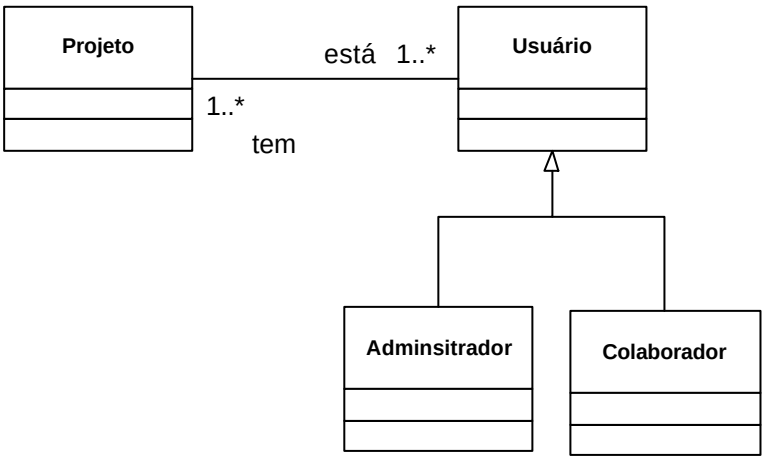


Figura 6.6 Diagrama de Classes do Gerenciamento de Usuários

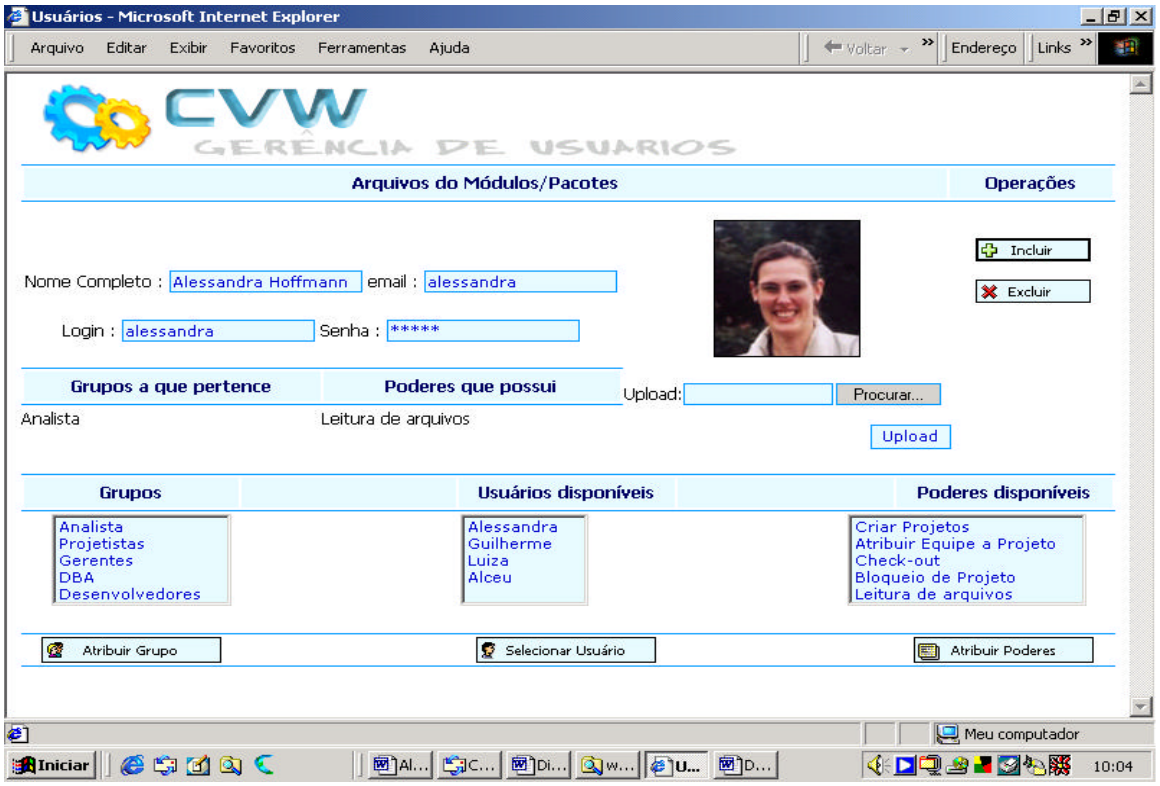
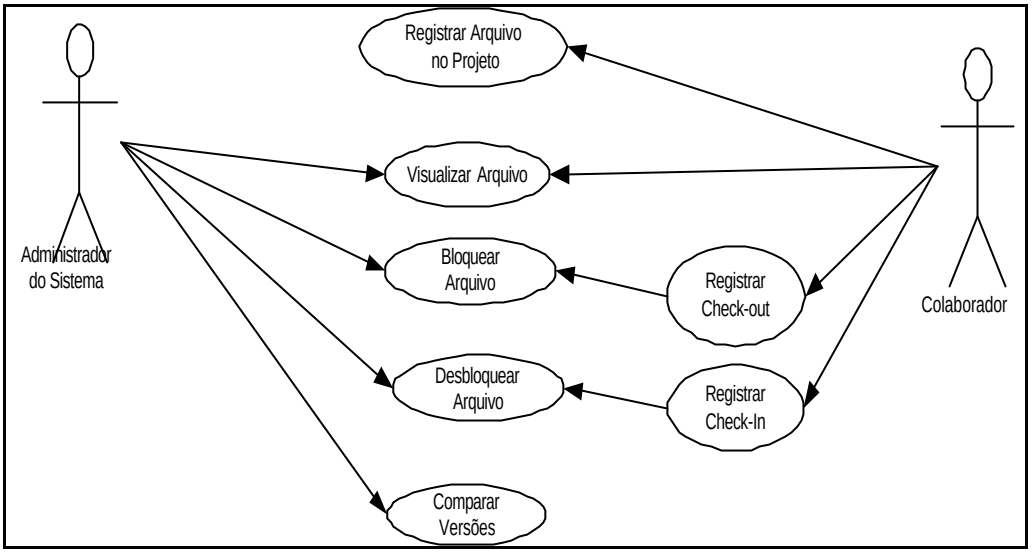


Figura 6.7 Tela Gerenciar Usuários - CVW

Gerenciar projeto

Para o gerenciamento de alterações de um ou mais arquivos do projeto o colaborador ou o administrador após selecionar o projeto que estará centralizado em um computador e visível através de uma árvore de diretórios, escolhe a última versão do arquivo e realiza a solicitação da operação de Check-out. Neste momento deve ser realizada uma cópia para a máquina do colaborador solicitante, sendo o arquivo "bloqueado" no servidor central até que seja efetuado seu Check-In (devolução) e "desbloqueado" (Figura 6.8 a Figura 6.10). A interface neste caso é toda realizada através do browser através da máquina envolvida no projeto (Figura 6.11).

Ainda, quando ocorrer o Check-in (devolução – Figura 6.9) é preciso antes fazer um backup do arquivo relacionado que está no servidor central e copiar a nova versão atualizando a página de dados.



Caso de Uso	Registrar Check-In
Atores	Colaborador
Finalidade	Registrar devolução do arquivo de um Projeto
Descrição	Este caso de uso é inicializado pelo COLABORADOR. Ele provê a funcionalidade de solicitar um arquivo de um projeto. Quando é realizado o Chek-In, automaticamente é realizado uma cópia para máquina do colaborador solicitante e bloqueado o arquivo na máquina central.
Tipo	Primário
Referência Cruzada	Registrar Projetos, Registrar Colaboradores

Caso de Uso	Realizar Check-Out
Atores	Colaborador
Finalidade	Registrar solicitação do arquivo de um Projeto
Descrição	Este caso de uso é inicializado pelo COLABORADOR. Ele provê a funcionalidade de devolução de um arquivo de um projeto. Quando é realizado o Chek-Out, automaticamente é realizado uma cópia para máquina central como uma nova versão. O arquivo na máquina central fica desbloqueado e a cópia do colaborador solicitante é bloqueado.
Tipo	Primário
Referência Cruzada	Registrar Projetos, Registrar Colaboradores

Figura 6.8 Diagrama de Use-Case do Gerenciamento de Projeto

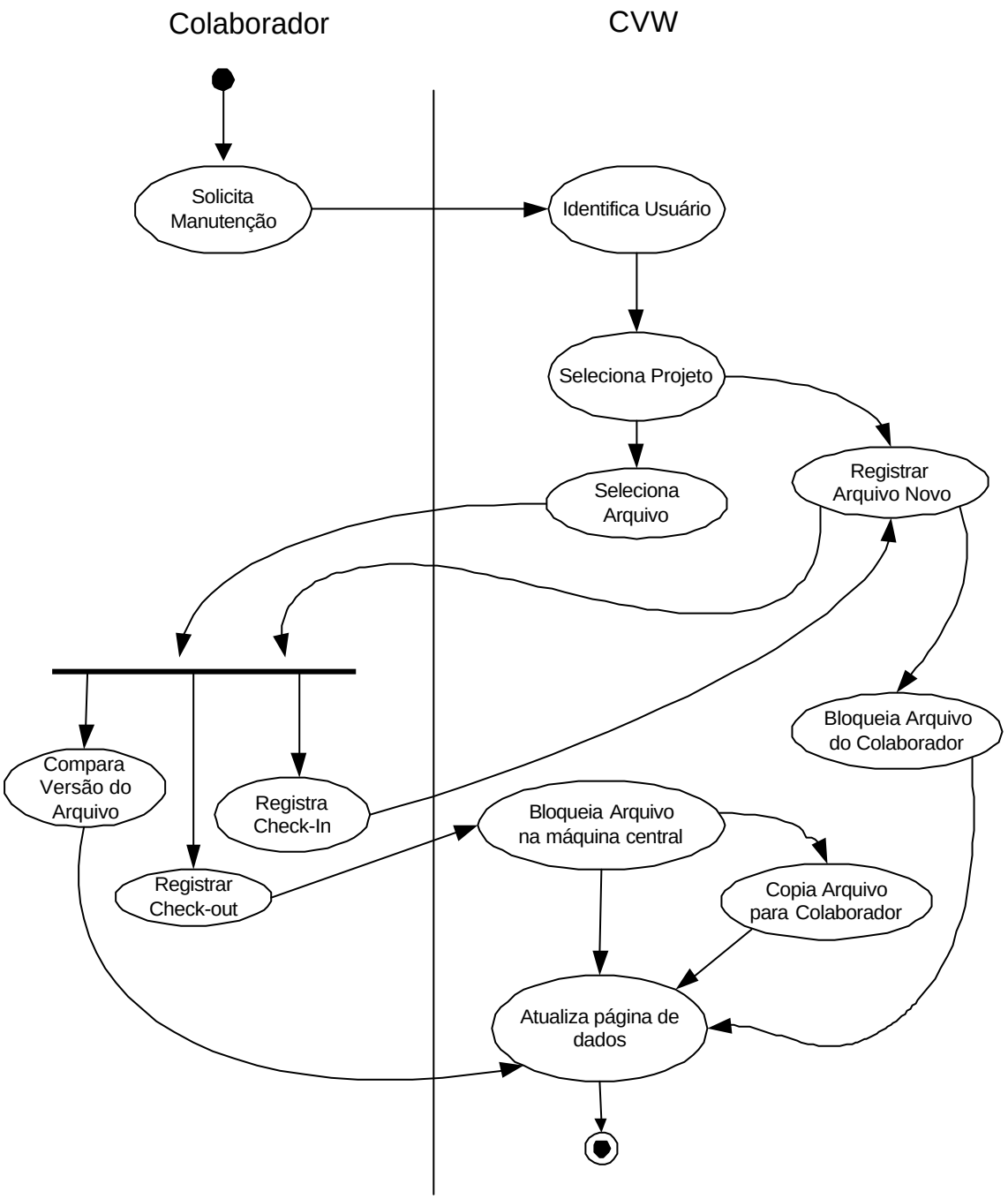


Figura 6.9 Diagrama de Atividades do Gerenciamento de Projeto – visão do Colaborador

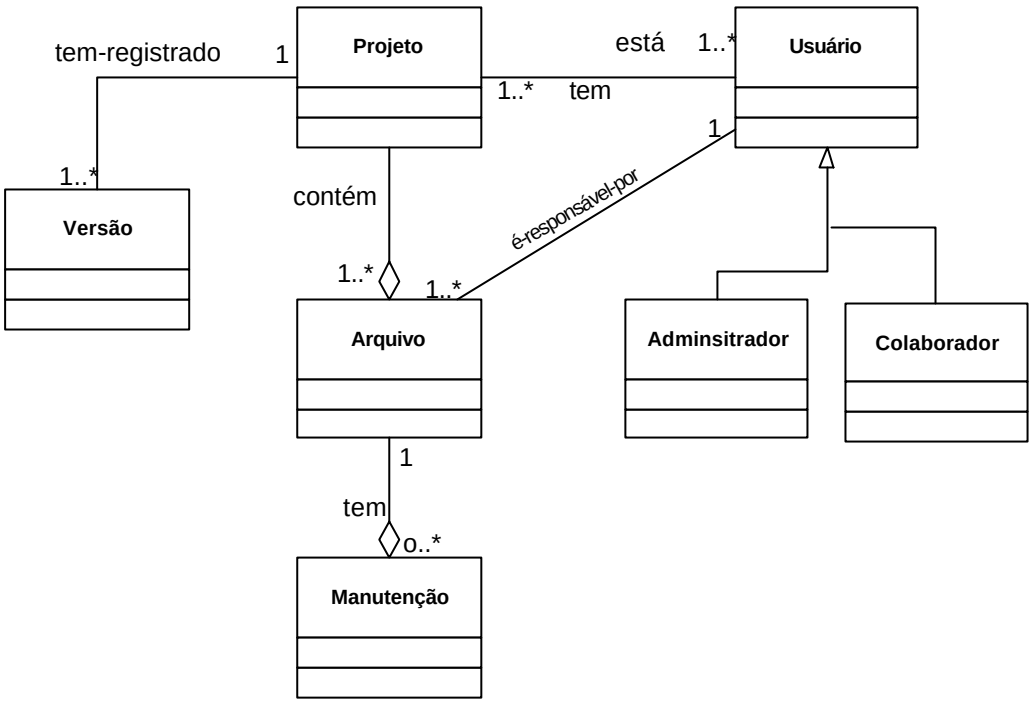


Figura 6.10 Diagrama de Classes do Gerenciamento de Projeto

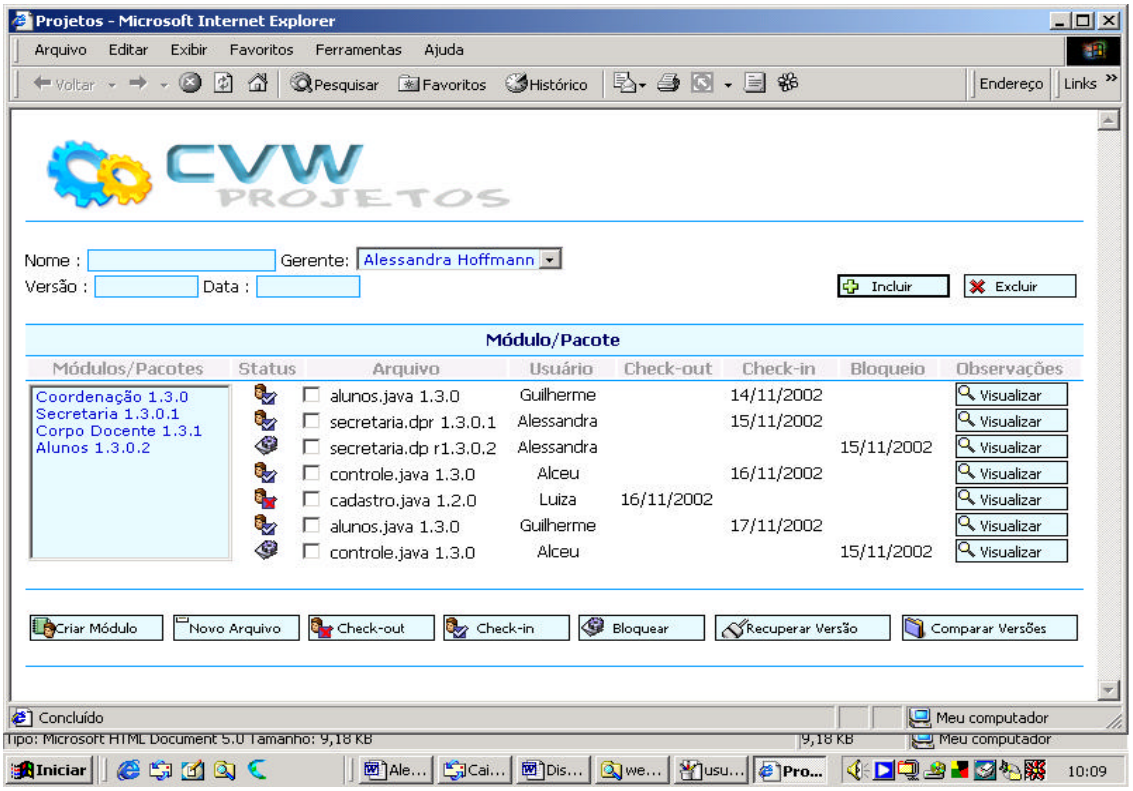
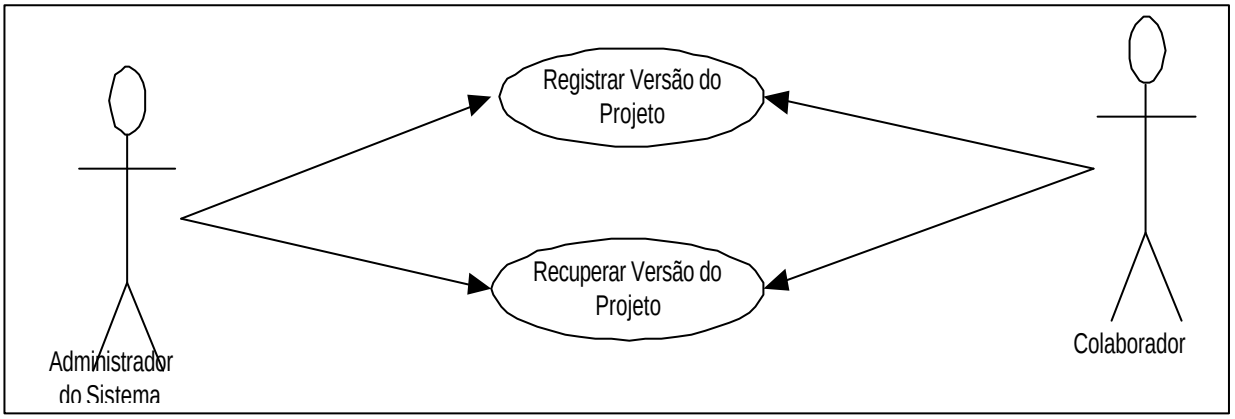


Figura 6.11 Tela de Gerenciar Projetos CVW

É disponibilizado um mecanismo que quando é realizada uma operação de Check-out, Check-in, ou inserção de um arquivo, esse pode ser avisado aos outros colaboradores. Assim, caso algum desenvolvedor esteja esperando a liberação de um arquivo para realizar alguma manutenção será avisado imediatamente.

Gerenciar versões

Verifica-se que este gerenciamento faz parte do gerenciamento do projeto. O administrador e o colaborador podem recuperar qualquer versão do sistema bem como atualizar uma versão, seja do arquivo como do projeto.



Caso de Uso	Registrar versão do Projeto
Atores	Administrador, Colaborador
Finalidade	Registrar nova versão do Projeto
Visão Geral	Este caso de uso é inicializado pelo ADMINISTRADOR ou COLABORADOR. Ele provê a funcionalidade de atualizar os dados de uma versão de um Projeto previamente selecionado.
Tipo	Secundário
Referência Cruzada	Registrar Projetos, Registrar Colaboradores, Registrar Arquivo no Projeto, Registrar Check-In

Caso de Uso	Recuperar versão do Projeto
Atores	Administrador, Colaborador

Finalidade	Recuperar outra versão do Projeto
Visão Geral	Este caso de uso é inicializado pelo ADMINISTRADOR ou COLABORADOR. Ele provê a funcionalidade de recuperar os dados de um Projeto previamente selecionado.
Tipo	Secundário
Referência Cruzada	Registrar Projetos, Registrar Colaboradores, Registrar Arquivo no Projeto, Registrar Check-Out

Figura 6.12 Diagrama de Use-Case do Gerenciamento de Versões

Para atualizar um arquivo é realizada uma busca das últimas versões existentes no repositório, sendo que o diretório de trabalho para onde o arquivo é copiado é determinado pelo colaborador .

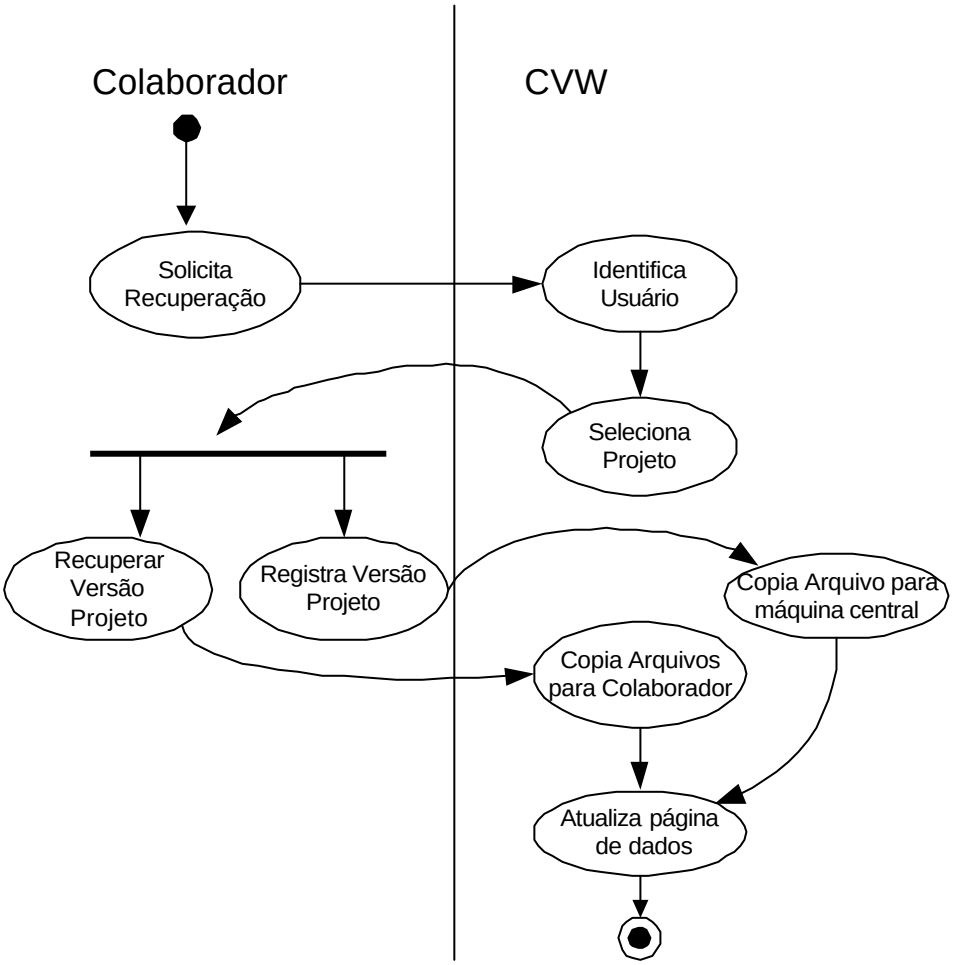


Figura 6.13 Diagrama de Atividades do Gerenciamento de Versões

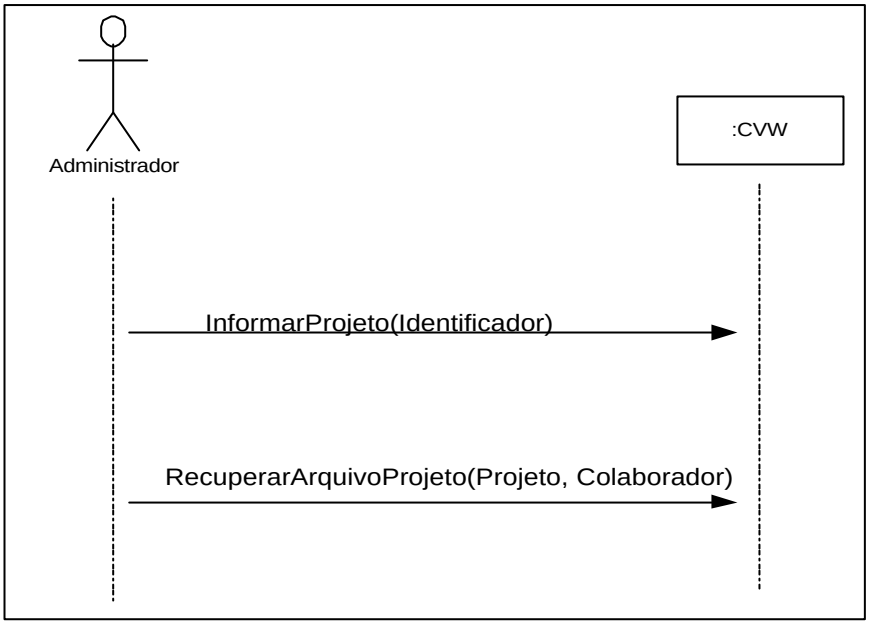


Figura 6.14 Diagrama de Seqüência do Gerenciamento de Versões

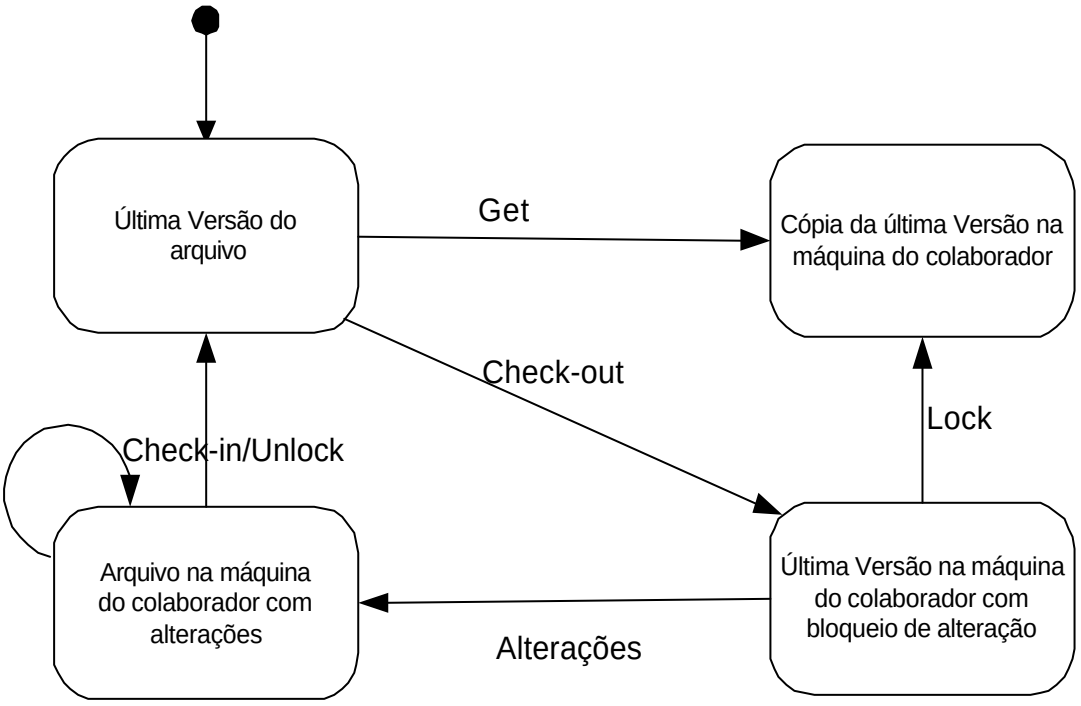


Figura 6.15 Diagrama de estado do Controle de Versão do arquivo

Foi também acrescentada a opção de comparação entre os arquivos, onde então é possível o desenvolvedor ou o administrador visualizar as diferenças entre dois ou mais arquivos (Figura 6.16).

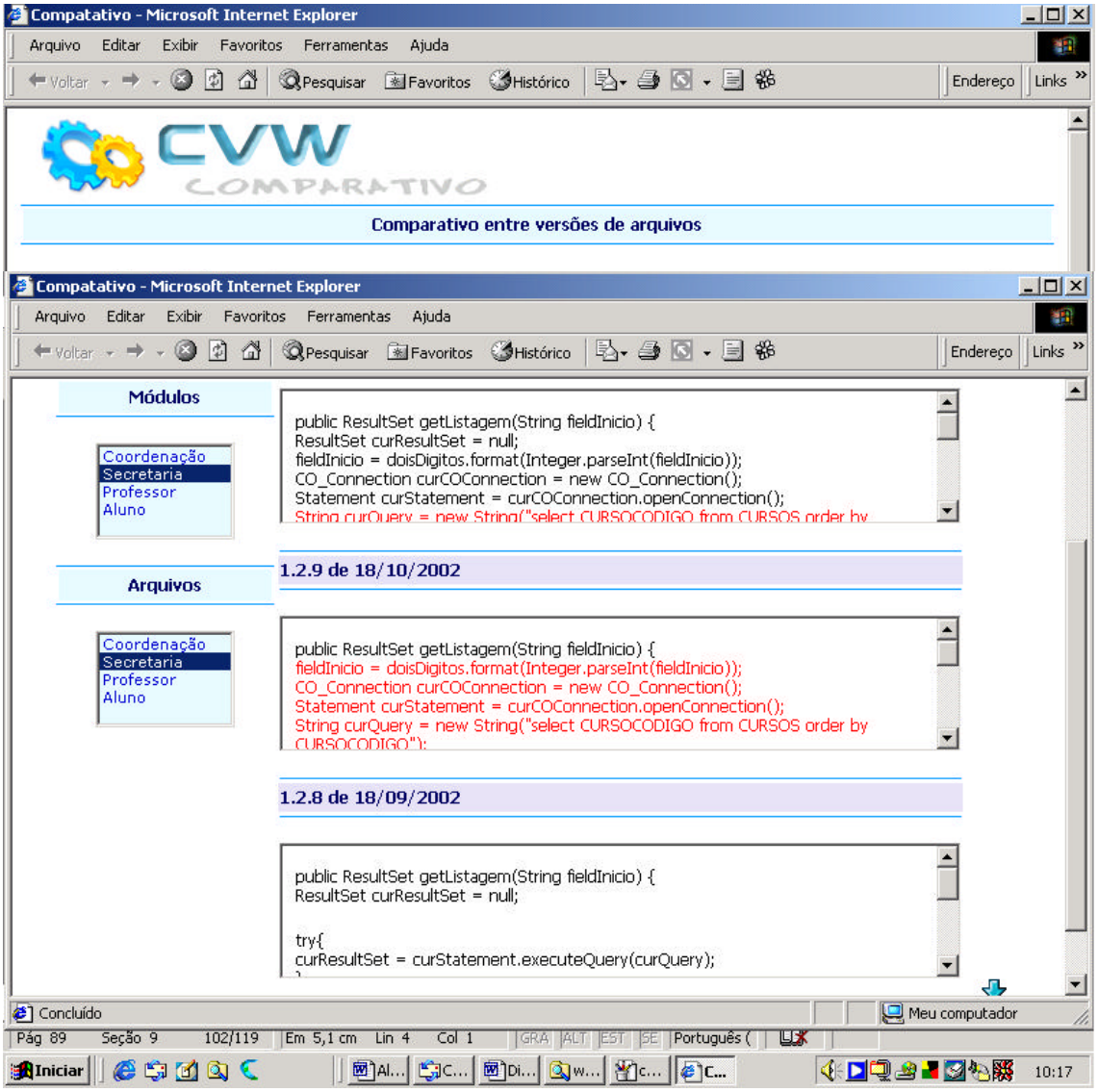


Figura 6.16 Tela Gerenciar Versão

6.4 Do controle de versões e políticas de manutenção

Para que um grupo de desenvolvedores em locais remoto esteja trabalhando em um sistema é preciso que haja um controle sobre as versões. Não utilizar um controle de versões ou ainda utilizar-se de forma incorreta, pode criar problemas que prejudiquem o cronograma do projeto e a qualidade do trabalho (HAVEWALA, 1999).

Sendo o gerenciamento do controle de versões a principal atividade do sistema, é importante verificar a necessidade de tratar as operações de bloqueio de arquivos quando este estiver sido requisitado por algum membro da equipe, já que não se pode deixar com que dois ou mais membros possam estar realizando alterações no mesmo arquivo ao mesmo tempo. Neste caso o arquivo do servidor central é marcado como somente para leitura, ficando bloqueado para retirada (check-out), sendo liberado assim que o membro que o bloqueou, faça o check-in do arquivo, desbloqueando o mesmo. Cada vez que isto ocorrer, uma nova versão do arquivo é armazenada, permanecendo as anteriores para posterior recuperação (Figura 6.3).

Cada desenvolvedor então tem permissão para alterar qualquer arquivo do projeto considerando a situação acima mencionada. Já o administrador deverá ter permissão para realizar todas as operações no sistema.

6.5 Histórico de Alterações

Quando um colaborador efetua alterações em um arquivo de um projeto, será armazenado em um banco de dados as informações sobre estas atualizações. Estes dados propiciam um importante histórico para a solução de problemas decorrentes da manutenção dos arquivos. Através deste histórico obtém-se informações pertinentes às alterações efetuadas, agilizando o acesso a dados como : desenvolvedor que efetuou a manutenção, data e hora em que ocorreram e a descrição informada.

6.6 CONCLUSÃO

A motivação para elaboração deste trabalho vão das dificuldades encontradas por grupos de trabalho no desenvolvimento de projetos, onde os membros, podem estar geograficamente distribuídos. Assim, o objetivo deste capítulo foi desenvolver um modelo de uma ferramenta para auxiliar no gerenciamento de versões de projeto por meio da própria Web, apoiando os colaboradores ou desenvolvedores de um projeto no trabalho colaborativo, sem que haja perda ou sobreposição de informações. Com a arquitetura adotada do repositório centralizado no servidor, facilita o processo de consistência, e, as versões se tornam disponíveis aos autores (colaboradores) à medida que cada um modifica o arquivo e faz o check-out . O fato do bloqueio do arquivo gera um processo de serialização, portanto demora ou perda de produtividade. Mas isto é necessário para que a qualidade tão desejada em desenvolvimento de software seja alcançada.

A princípio pode-se observar uma contribuição inicial deste trabalho: a viabilização de uma ferramenta de controle de versões na Web, através de um ambiente conhecido e praticamente com mínimo de implantação (o usuário precisa apenas de um browser Web em sua máquina local para utilizar a ferramenta, favorecendo ainda mais a sua mobilidade). Em ambientes de engenharia de software e ambientes de autoria, uma das vantagens encontradas mais significativas está relacionada com a reconstrução de versões anteriores de seu trabalho.

Capítulo 7

Conclusões e Perspectivas para Futuros Trabalhos

A sociedade onde vivemos sofreu algumas mudanças desde que a globalização se tornou não somente um termo, mas uma realidade perante o mercado. Dizemos que com a Internet não há mais distância física, e isto é uma realidade, então aplicações na Web podem ser vistas como um novo tipo ambiente para interação.

Existe um crescimento e melhora da produtividade em algumas atividades, tais como o desenvolvimento de um projeto de software, quando a equipe trabalha em regime de colaboração possibilitando com que o desenvolvimento de projeto, em específico o software, seja maior do que a soma das partes, visto que o limite geográfico não é mais uma barreira.. De fato, o mecanismo mais eficiente é aquele que consegue unir o trabalho humano com uma ferramenta automatizada. Portanto se pudermos unir em uma ferramenta que gerencie e auxilie os profissionais de desenvolvimento de software a Internet e a colaboração, teremos um produto final com maior qualidade .

A análise de ferramentas colaborativas mostrou que os conceitos de colaboração e Internet ainda são pouco explorados, sendo este um ponto de partida para sua implementação, e futuras pesquisas de aplicações.

No modelo de ambiente proposto -que utiliza o ambiente colaborativo distribuído- quando um arquivo é solicitado (Chek-out) o bloqueio deve ser automático. Isto faz com que o arquivo solicitado, não possa ser editado por outro colaborador. Esta política pode ser aplicada com o uso do CVS. Isto é perfeitamente possível, pois o acesso simultâneo sobre um mesmo arquivo pode ser alterado em suas configurações. Outro fator é que trabalhando com o CVS e a interface Web, os colaboradores não precisam saber quais as linhas de comando do CVS fazem tal operação. Assim a ferramenta tem mais uma facilidade para o uso colaborativo distribuído, e faz com que os colaboradores possam aprofundar seu conhecimento no desenvolvimento do projeto.

A centralização das versões no servidor, facilita este processo, pois estas estarão disponíveis a todos colaboradores. Com o recurso de comparação de arquivos, juntamente com observações da versão gerada faz com que o processo tenha maior agilidade. As vantagens do ponto de vista da engenharia de software é a reutilização das versões anteriores, além da localização das informações tais como as alterações, autores, data, entre outros, relevantes para histórico do desenvolvimento.

A relevância de o projeto ter um administrador responsável, em caso de surgir algum problema por parte do colaborador, faz com que aquele possa bloquear ou desbloquear qualquer arquivo. Desta forma a produtividade não é comprometida.

Sugestão para trabalhos futuros

Como sugestão de trabalhos futuros, visando um aperfeiçoamento e um aprofundamento indica-se uma ampliação nas políticas de manutenção, criando-se mecanismos de controle adaptáveis a cada equipe de trabalho, como tempo de bloqueio, prazos de atualização a serem cumpridos, entre outros. Mecanismo interessante pode ser o que amplia as políticas de acesso, procurando fazer a integração do cliente (empresa), onde esse poderia realizar update ou download da versão do projeto.

E finalmente a implementação do modelo, aplicação e testes que resultariam em mais uma contribuição para o apoio ao gerenciamento de desenvolvimento de projeto de software.

ANEXO 1

O que é apresentado neste documento é parte de uma primeira e até o momento única pesquisa que foi realizada em 1999 de abrangência nacional sobre a utilização de Gerência de Configuração de Software como aprofundamento dos levantamentos sobre qualidade realizados bianualmente no âmbito do Sub-Programa Setorial da Qualidade e Produtividade em Software, do Programa Brasileiro da Qualidade e Produtividade – SSQP/SW – PBQP[16]. Segundo a Fundação Centro Tecnológico para Informática– CTI com apoio da Secretaria de Política de Informática e Automação – SEPIN, órgãos do Ministério da Ciência e Tecnologia, esta pesquisa resultou nas seguintes informações :

Na questão levantada quanto ao conhecimento do termo "gerência, gerenciamento ou gestão de configuração de software". Das empresas pesquisadas, 21,2% declaram não ter nenhum conhecimento sobre esses termos e 56,1% declaram ter um conhecimento superficial. Apenas 22,7% das empresas conhecem bem o assunto (Tabela 1).

Quanto ao conhecimento relativo ao termo **Controle de Versões,** 4,5% das empresas declaram não saber do que se trata e 25,8% declaram que sabe do que se trata, mas nunca utilizou. As demais empresas, 69,7% declaram que sabe do que se trata e já utilizou, conforme pode ser observado na Tabela 2.

---

16 O PBQP é um programa do governo brasileiro, implantado em 1990, visando a modernização do processo de produção através da promoção da qualidade e produtividade, com conseqüente ampliação da competitividade de bens e serviços produzidos no País. A operacionalização do PBQP apóia-se, fundamentalmente, na iniciativa e recursos próprios dos agentes econômicos e nos meios disponíveis nos organismos governamentais

Tabela 1 Conhecimento Relativo ao Termo "Gerência/Gerenciamento/Gestão de Configuração de Software"

Grau de conhecimento	Nº de empresas	%
Nenhum	75	21,2
Conhecimento superficial	198	56,1
Conhece bem o assunto	80	22,7
Total	***353***	***100***

Obs: 2 empresas não responderam a questão

Fonte : **Pesquisa GCS -Gerência de Configuração de Software.** ***Fundação Centro Tecnológico para Informática - CTI / IC . Programa de Qualidade e Produtividade em Software – PQPS. Área de Tecnologia para Produção de Software – ATPS. ? ?Fundação CTI 1999***

Tabela 2: Conhecimento Relativo ao Termo "Controle de Versões".

Grau de conhecimento	Nº de empresas	%
Não sabe do que se trata	16	4,5
Sabe do que se trata, mas nunca utilizou	91	25,8
Sabe do que se trata e já utilizou.	246	69,7
Total	***353***	***100***

Fonte : **Pesquisa GCS -Gerência de Configuração de Software.** ***Fundação Centro Tecnológico para Informática - CTI / IC . Programa de Qualidade e Produtividade em Software – PQPS. Área de Tecnologia para Produção de Software – ATPS. ? ?Fundação CTI 1999***

Em se tratando da adoção de ferramentas para automatizar a gerência de configurações/controle de versões do software, 35,9% das empresas afirmam que adotam algum tipo de ferramenta. No entanto, 64,1% das empresas não adotam ferramentas, de acordo com a Tabela 3.

Das 126 empresas que afirmam utilizar ferramentas para automatizar os procedimentos de gerência de configuração/controle de versões do software, 123 especificaram o tipo de ferramenta utilizada. Dessas empresas, observa-se na Tabela 4 que, 45,5% desenvolvem suas próprias ferramentas para esses procedimentos e 44,7% adotam ferramentas comerciais. Para 9,8% das empresas a gerência é automatizada pela ferramenta do próprio software de desenvolvimento.

Tabela 3: Adoção de Ferramentas para Automatização da Gerência de Configurações /Controle de Versões de Software.

Abordagem	*Nº de empresas*	*%*
Adota ferramentas	**126**	**35,9**
Não adota ferramentas	**225**	**64,1**
Total	***351***	***100***

obs: 4 empresas não responderam a questão

Fonte : **Pesquisa GCS -Gerência de Configuração de Software.** ***Fundação Centro Tecnológico para Informática - CTI / IC . Programa de Qualidade e Produtividade em Software – PQPS. Área de Tecnologia para Produção de Software – ATPS. ? ?Fundação CTI 1999***

Tabela 4: Distribuição das Empresas por Tipos de Ferramentas Adotadas na Gerência de Configurações /Controle de Versões de Software.

Abordagem	***Nº de empresas***	***%***
Ferramenta desenvolvida pela empresa	**56**	**45,5**
Ferramenta comercial	**55**	**44,7**
Ferramenta do software de desenvolvimento	**12**	**9,8**
Total	***123***	***100***

obs: 3 empresas não responderam a questão

Fonte : **Pesquisa GCS -Gerência de Configuração de Software.** ***Fundação Centro Tecnológico para Informática - CTI / IC . Programa de Qualidade e Produtividade em Software – PQPS. Área de Tecnologia para Produção de Software – ATPS. ? ?Fundação CTI 1999***

Glossário

BRANCH - um desvio; uma instrução de desvio no programa que podem ser seguidos; ir de uma parte de um programa para outra. Neste caso, cada versão origina outra que chamará sua sucessora ou vice-versa;

BROWSER - programas de aplicação específicos, com capacidade de comunicação com os servidores e capazes de enviar solicitações para busca de documentos através da Internet.

CLIENTE/SERVIDOR - Um sistema, chamado de servidor, fornece um recurso em particular. O outro sistema, chamado de cliente, faz o uso deste recurso e serviço para o compartilhamento de informações.

CMM - *Capability Maturity Model* – definido no Software Engineering Institute (SEI) – Carnegie Meloon University . define o nível de maturidade de uma empresa quanto ao desenvolvimento dos processos dentre 5 estabelecidos. Tem como objetivo auxiliar as organizações a aumentarem a maturidade de seu processo por um caminho evolutivo.

CSCW - *Computer-Supported Cooperative Work* - Trabalho Cooperativo Suportado por Computador - pesquisa para o estudo das técnicas e metodologias, e as formas que a tecnologia auxilia no trabalho em grupo;

GROUPWARE – O groupware engloba um conjunto de tecnologias especialmente vocacionadas para tornar os grupos mais produtivos. Mais especificamente, trata-se de software que apoia a cooperação de grupos de pessoas, permitindo mesmo que estas trabalhem eventualmente em ambientes distribuídos.

GSS – *Group Suport System* – ferramenta com objetivo de aumentar a produtividade de reuniões, acelerando o processo de tomar decisões ou melhorando a qualidade das decisões resultantes.

HOST – Na Internet, cada computador separado é chamado host. Ou seja, para você obter uma informação de qualquer lugar, você conecta-se a um host, se fosse entrar num site da França, por exemplo, o seu computador se conectaria a um host onde está hospedado o site. E se o seu computador estiver conectado à Internet, ele também é um host, mesmo não acessando qualquer recurso disponível na Internet.
Mas esse não é o único conceito de host, o outro pode ser distinguido como nó. Dentro de um desenho de uma determinada rede existem as conexões entre as junções, cada computador será um ponto e cada conexão será uma linha. Matematicamente falando cada uma dessas junções ou pontos são chamados de nós. Por isso, o termo host em português seria "nós", que seria para designar qualquer computador que esteja conectado a uma rede

ISO 9000 - série de certificação- conjunto de normas que tratam de sistemas da qualidade do processo de software. Demonstra que o sistema de Qualidade da Organização é efetivo . Esta certificação tem prazo de validade, sendo necessário revisão e não avalia diretamente a qualidade de nenhum produto ou serviço.

LAN - *Local Area Network*- é uma rede local, ou seja, são computadores conectados diretamente, geralmente por algum cabo. Existe também as WANs (Wide Area Network) que são grupos de redes LANs interligadas. Essas conexões remotas (WAN) são interligados nomalmente por linhas telefônicas ou por outra tecnologias tipo: satélites, ondas, etc

MULTICAST - transmissão para um número de receptores ou nós (computadores), com um endereço em cada mensagem para indicar o nó desejado.

ROOT - nó inicial a partir do qual todos os caminhos se derivam em uma estrutura de árvore de dados; Diretório raiz (principal), que pode conter outros subdiretórios;

SCM – *Software Configuration Management*, o mesmo que Gestão de Configuração de Software, atividade aplicada a todo o processo da Engenharia de Software e que gerencia a evolução de sistemas de software.

TCP/IP - Transmission Control Protocol over Internet Protocol- O protocolo TCP/IP é um conjunto de mais de 100 protocolos que são usados para conectar computadores e redes. Dentro da rede Internet os dados são transmitidos de um host para outro como um fluxo constante. Pelo contrário, eles são fragmentados em pequenos pedaços chamados pacotes. Resumindo, o IP tem o trabalho de transportar os dados no estado bruto de um lado para outro. Enquanto o TCP cuida do fluxo e da integridade dos dados

WHITEBOARDING – Ferramenta de Conferência de dados utilizado em salas de conferência. O computador conectado capta tudo o que se escreve no whiteboard (rascunho de uso comum) e grava em um arquivo. Pode-se projetar documentos e arquivos de desenho no whiteboard de modo que os participantes da reunião possam fazer mudanças e salva-las em arquivo. Outro tipo é que conecta dois ou mais computadores via telefone. Neste caso, todos os participantes podem compartilhar documentos e arquivos nos dois computadores e marca-los com revisões. Neste caso, cada localidade pode imprimir uma cópia para registrar os resultados da reunião. Essas ferramentas funcionam bem para criação, revisão de relatórios, apresentações, planos de projeto, orçamentos e projetos de produto.

WORKFLOW – ferramentas que executam atividades estruturadas com base em um conjunto de regras que governam o fluxo de documentos ou de formulários. Ele simplesmente é um automatizador de processos de negócios. Essas ferramentas utilizam banco de dados de correio eletrônico ou de documentos para enviar a informação para onde precisa ir.

WWW - *World Wide Web*. "Teia de alcance mundial". Baseada em hipertextos, integra diversos serviços Internet que oferecem acesso, através de hiperlinks, a recursos multimídia da Internet.

Referências Bibliográficas

ALLAN,G.W. **Configuration Management and its Impact on Businesses that use Computer Platforms** - International Journal of Project Management, Vol. 15 No 5, 1997. site http://www.dis.port.ac.uk/~allangw/papers/pub97d.htm acessado em 20/10/2002.

ARNSON,Bob **. CVS for news users** http://www.cvshome.org/new_users.html acessado em 28/06/2002

BLANDY, Jim . **Introduction in CVS**. http://www.cvshome.org/docs/blandy.html acessado em 28/06/2002

BOUNDS,Nadine M. e Susan Dart. **CM Plans: The Beginning to your CM Solution**. The Software Engineering Institute (SEI). Modificado em novembro de 2001. site http://www.sei.cmu.edu/legacy/scm/papers/CM_Plans/CMPlans.Chapter1.html#RTFToC3 acessado em 01/11/2002.

BURTON SYSTEM SOFTWARE, site na Internet em http://www.burtonsys.com/ , acessado em 29/08/2002.

CIAO, http://www.gaia.inf.br/web%5Cgaiahome.nsf/TeamStudioCiao?OpenPage site acessado em 25/07/2002.

CANIZARES, Osvaldo. **Trabalho colaborativo em Projetos de Construção Civil através da Internet.** Dissertação de Mestrado – UFSC- Universidade Federal de Santa Catarina. Florianópolis, p. 30-47, 2001.

CONRADI R., B. Westfechtel, **Version Models for Software Configuration Management**, ACM Computing Surveys, vol. 30, Nº. 2, 1998.

CVS , http://www.cvshome.org/project/www/docs/ddSourceBrowse.html acessado em 28/06/2002.

DELGADO, Armando Luiz Nicolini .**Tendências em Protocolos de Aplicação para Ambientes Colaborativos Distribuídos**. Julho 1998. Arquivo acessado em

14/05/2002 do site http://www.dca.fee.unicamp.br/courses/IA368F/1s1998/Monografias/armando/

ELLIS, C. A., GIBBS, S. J. and REIN G. L. **Groupware: Some Issues and Experiences. Comm**. of the ACM, 34(1): 38-58. Jan. 1991.

FLUCKIGER, F. **Understanding networked multimedia**. Prentice Hall, 1995.

GCS, **Gerência de Configuração de Software - Pesquisa GCS** . Fundação Centro Tecnológico para Informática - CTI / IC . Programa de Qualidade e Produtividade em Software – PQPS. Área de Tecnologia para Produção de Software – ATPS. Fundação CTI, 43p.,1999.

GROSZ, B. J. **Collaborative Systems**. AI Magazine, 17(2):67-85. Summer 1996.

HAVEWALA, Aspi. **The version control process**. Dr. Dobb's Journal. São Francisco,n299,p100-111, maio de 1999.

HENDERSON, Ron. **"Version Control for Web Development using CVS".** ArsDigita Systems Journal. 08/Jun/2000.

HILLS, Mellanie. **Intranet como groupware.** Trad. Luis Alfonso Sanchez Balaguer. São Paulo, Berkeley Brasil, p.7-81,1997.

ICARUS. **"Version Controle With CVS",** November 2000. http://www.devshed.com acessado em 29/06/2002.

IEEE 610.12-1990. **IEEE Stand for Software Configuration Management Plans (ANSI)**. The Institute of Eletrical and Eletronics Engineers Psicataway, New Jersey, 83p.

IEEE 828-1990. **IEEE Stand for Software Configuration Management Plans (ANSI)**. The Institute of Eletrical and Eletronics Engineers Psicataway, New Jersey, 16p.

INFORMATIVO.**Informativo Técnico nº 71- Sistemas colaborativos** (março 2000) . Acessado em 29/04/2002 do site http://www.revista.unicamp.br/infotec/informação/inf712.htm

KLING, R. **Cooperation, Coordination and Control in Computer-Supported Work**. Comm. of the ACM, 34(12):83-88. Dec. 1991.

LOZANO, Fernando **. Software Livre Para o Desenvolvedor.** Revista do Linux, http://www.revistadolinux.com.br/ed/023/assinantes/desenvolvimento.php3 , acessado em 09/11/2002.

MARK, C. Chu-Carroll, Sara Sprenkle, Coven. **Brewing better collaboration throught software configuration management**. ACM SIGSOFT Engineering Notes, v.25 n6, p. 88-97, November 2000.

MAZZOLA, V.B. **"Internet e Intranets"**. Apostila da disciplina de Pós-Graduação. INE/UFSC, 2001.115p. Florianópolis, SC.

MERANT, site na Internet em http://www.merant.com/products/pvcs/vm/index.asp , acessado em 22/08/2002.

MICROSOFT CORPORATION, site na Internet em http://msdn.microsoft.com/ssafe acessado em 22/06/2002

MICROSOFT CORPORATION, site na Internet em http://www.microsoft.com/catalog/display.asp?subid=40&site=606 acessado em 22/07/2002

ORAM, Andy . **Peer-to-Peer: o poder transformador das redes ponto a ponto**.[tradução Bazán Tecnologia e Lingüística]. – São Paulo: Berkeley Brasil,p.3-22 2001.

PAULA, Wilson de Pádua. **Engenharia de Software – Fundamentos, Métodos e Padrões**. Rio de Janeiro, LCT. 2001

PAAS, Leslie Christine. **A Integração da Abordagem Colaborativa à Tecnologia Internet para Aprendizagem Individual e Organizacional no PPGEP** .Dissertação de Mestrado – UFSC- Universidade Federal de Santa Catarina. Florianópolis, 1999

PENG, C. **Survey of Collaborative Drawing Support Tools: Design Perspectives and Prototypes**. Computer Supported Cooperative Work, 1(3): 197-228. 1993.

PRESSMANN, Roger S., **Engenharia de Software** - McGraw-Hill, Inc, 1995. 3º ed.

PRESSMANN, Roger S., **Software Engineering : A Practitioner's Approach** - McGraw-Hill, Inc, 1997. 4º ed.

QUALITY SOFTWARE COMPONENTS, site na Internet em http://www.qsc.co.uk/gpv/gpversion.htm , acessado em 29/08/2002.

RADDING, Alan . **Join The Team .** Information Week On-line. Publicado em 4 de outubro de 1999. site http://www.informationweek.com/755/55adtea.htm acessado em 20/10/2002.

RAPOSO, Alberto B., Léo P. Magalhães e Ivan L. M. Ricarte **. Interação na Web.** Acessado do site http://www.dca.fee.unicamp.br/~alberto/pubs/JAI99/curso_jai99.html em 19/10/2002.

RCS http://www.csc.calpoly.edu/~dbutler/tutorials/winter96/rcs/ site acessado em 27/09/2002.

RATIONAL ClearCase, http://www.rational.com , acessado em 27/09/2002 .

REINHARD, W., SCHWEITZER, J. et al. **CSCW Tools: Concepts and Architectures**. IEEE Computer, 27(5): 28-36. May 1994**.**

RELIABLE SOFTWARE, site na Internet em http://www.relisoft.com/co_op/ , acessado em 27/08/2002.

REZENDE, Denis Alcides. **Engenharia de software e sistemas de informações.** Rio de Janeiro: Brasport, 1999.

RIZZO, Thomas . **An Introduction to Collaborative Systems from Programming Microsoft Outlook and Microsoft Exchange**. Arquivo acessado em 09/05/2002 do site http:\\www.microsoft.com/technet/treeview/default.asp?url=/technet/prodtechnol/exchange/plan/collabso.asp

ROCHA, Ana Regina Cavalcanti da. **Qualidade de software** . Apostila do Curso de Especialização em Engenharia de Software. UNIOESTE –ITAI – COPPE/UFRJ, 100 p.1998. Foz do Iguaçu, PR.

ROCHA, Ana Regina Cavalcanti da. **Qualidade de software** . São Paulo: Pretence Hall, 2001.

SCHMIDT, K. **Riding a Tiger, or Computer Supported Cooperative Work**. Proc. ECSCW/91, pp 1-16. 1991.

SCHMIDT, K. and BANNON, L. J. **Taking CSCW Seriously - Supporting Articulation Work. Computer Supported Cooperative Work**, 1(1-2): 7-40. 1992.

SHIRMOHAMMADI, S., de OLIVEIRA, J. C. and GEORGANAS, N. D. **Applet-Based Telecollaboration: A Network-Centric Approach**. IEEE Multimedia, 5(2): 64-73. Apr./Jun. 1998

SOARES, M. D.; R. P. M. Fortes. **Gerenciamento de Controle de Versões de Páginas Web**, Anais do VI Workshop de Teses e Dissertações defendidas do ICMC/USP, junho de 2001.

TICHY ,W. F. **RCS – A system for version control**, Software Practice and Experience, Vol. 15, Nº 7, 637-654, 1985.

YAVATKAR, R. and Lakshman, K.: **Communication support for distributed collaborative applications**. *Multimedia Systems*, 2:74-88, 1994.

YOURDON, Edward. **Análise Estruturada Moderna.** Rio de Janeiro: Campus, 1990. 3º ed.

WILSON, P. **Introducing CSCW - What It Is and Why We Need It**. In: SCRIVENER, S. A. R. (Ed.). Computer-Supported Cooperative Work - The multimedia and networking paradigm, pp 1-18. Unicom Seminars Ltd., 1994.